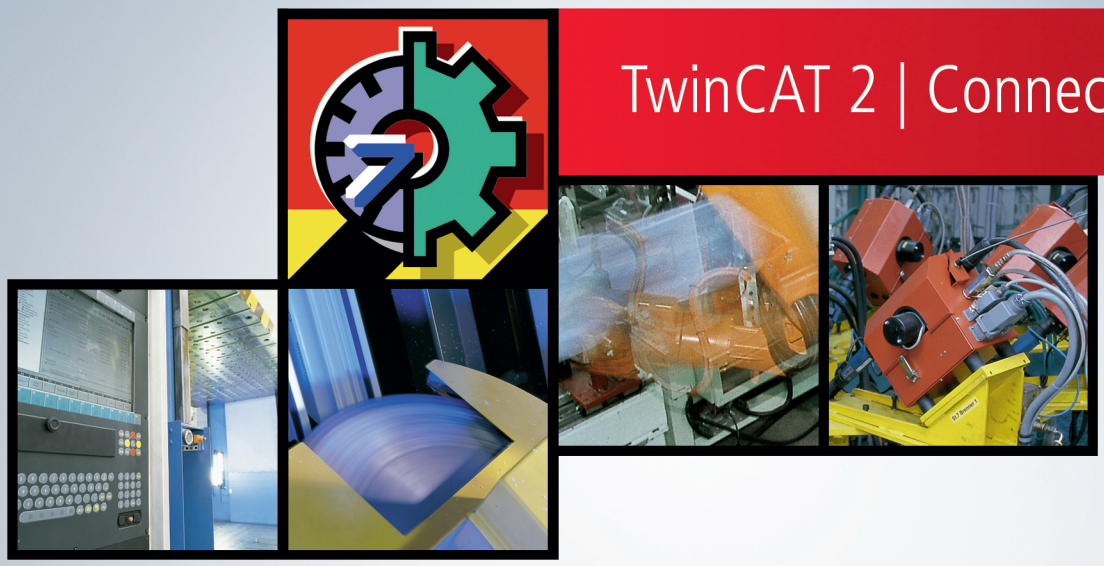
BECKHOFF New Automation Technology

Manual | EN

TX1000

TwinCAT 2 | ADS.NET



TwinCAT 2 | Connectivity

2023-01-23 | Version: 1.3

Table of contents

1 Foreword

1.1 Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with applicable national standards.
It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components.
It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without prior announcement.
No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.
Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
with corresponding applications or registrations in various other countries.

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

Copyright

1.2 Safety instructions

Safety regulations

Please note the following safety instructions and explanations!
Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

Exclusion of liability

All the components are supplied in particular hardware and software configurations appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation and drive engineering who are familiar with the applicable national standards.

Description of symbols

In this documentation the following symbols are used with an accompanying safety instruction or note. The safety instructions must be read carefully and followed without fail!

⚠ DANGER

Serious risk of injury!

Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.

⚠ WARNING

Risk of injury!

Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.

⚠ CAUTION

Personal injuries!

Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.

NOTE

Damage to the environment or devices

Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.

Tip or pointer

This symbol indicates information that contributes to better understanding.

1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

2 Version History

The topics in this section describe the various changes made to the 'TwinCAT.Ads .NET API' over the life of the project.

Select a version below to see a description of its changes.

Other Resources

TwinCAT ADS .NET API Documentation [▶ 16]

2.1 Version 4.3.X.X

Feature milestones of the version 4.3.X.X Series

Milestones of Version 4.3.0.0

- Support of runtime sized Array Instances (AnySizeArrayInstance)
- Support of Nuget Semantic Versioning support:

  Because now the Versioning of Nuget packages supports API Updates, it is not necessary anymore to update the TwinCAT.Ads.dll AssemblyVersion - which will force to recompile dependent assemblies. From Version 4.3.0.0 on the AssemblyVersion will change only on major updates, what means that minor Updates (within 4.3.X.X) will remain the same AssemblyVersion and remain backward compatible - only the AssemblyFileVersion will change.

  Nuget package versioning

  Semantic Versioning specification (Semver.org)

Other Resources

Version History [▶ 14]

2.2 Version 4.2.X.X

Feature milestones of the version 4.2.X.X Series

Milestones of Version 4.2.172.0

- Release of TwincAT ADS Reactive (Ads.Rx) Nuget package Beckhoff.TwinCAT.Ads.Reactive package on Nuget The classes of the Reactive Extensions can be found in the namespace TwinCAT.Ads.Reactive.

Milestones of Version 4.2.171.0

- Release of Nuget package Beckhoff.TwinCAT.Ads package on Nuget

Milestones of Version < 4.2.170.0

- Dedicated AdsClientSettings object.
- Better Support of jagged arrays.
- Support of 64-Bit types ULINT, LINT, LTIME, LWORD, UXINT
- Enhanced support of ANYTypes (primitive .NET Arrays)
- Implementation of SymbolLoaderFactory to create SymbolLoader V2.
- Api for ADS Sum commands within the TwinCAT.Ads.SumCommand namespace.
- RPC Methods (Remote procedure calls) implementation.
- Support of custom symbol providers.

Other Resources

Version History [▶ 14]

2.3 Version 4.1.X.X

Feature milestones of the version 4.1.X.X series

Milestones in this version series:

- Implementation of Symbol Loader V2
- Dynamic Language Runtime Support (DLR) for CLR4 Versions.
- Fail fast handler implementation on connection timeouts
- Session and Connection support.
- Extended support of DataTypes in Symbol loaders.
- Support of arrays with more than 3 dimensions.
- Recognition of WSTRING types in AdsSymbolParser corrected.
- Support of Pointer and Reference types.
- Support of of Symbol and DataType Attributes.
- ReferenceType Recursion recognition added.
- UINT64 Support for ReadAny.
- Enhanced Common Error handling.
- Extended support for PlcOpen DataTypes (TIME, DATE, TOD, DT)

Other Resources

Version History [▶ 14]

2.4 Version 4.0.X.X

Feature milestones of the version 4.0.X.X series

Major Milestones:

- Support for .NET Framework 4.0

Other Resources

Version History [▶ 14]

3 TwinCAT ADS .NET API Documentation

The TwinCAT .NET API implements support for the TwinCAT Automation Device specification (ADS). It can be used within .NET Framework programming languages and support ADS Client implementations. The ADS API interface permits:

- The Implementation of ADS Clients
- Browsing of (ADS) server side symbolic information.
- Reading and writing ProcessImage information in Raw or in type safe manner.
- Receiving ADS Notifications as events.

Getting Started

For getting started please have a look at the following documents:

- Prerequisites [▶ 17]
- Installation [▶ 17]
- Concepts [▶ 19]
- HowTo Samples [▶ 1981]

Background information about the TwinCAT ADS protocol can be found here:

- ADS Introduction
- TwinCAT ADS Device concept

for common ADS information.

From the conceptual standpoint within this ADS .NET API reference documentation, the most important starting points for reading are the following Classes/Methods:

Main documentation entry points

Description	Link
Addressing ADS Devices via AmsNetId Address information.	AmsNetId [▶ 428]
Communicate to ADS Devices via the TcAdsClient class.	TcAdsClient [▶ 625]
Browse target system symbolic information (new Version, V2). Creation of the SymbolLoader via Factory class.	SymbolLoaderFactory.Create(IConnection,ISymbolLoaderSettings) [▶ 1102]
Browse target system symbolic information (traditional Version, V1))	TcAdsClient.CreateSymbolInfoLoader() [▶ 693]
Session and Connection management on top of the ADS communication channel established by the TcAdsClient.	AdsSession [▶ 347], AdsConnection [▶ 148]
Usage of Ads.Rx (Reactive extensions) to write reactive code.	AdsClientExtensions [▶ 821], AdsClientExtensions [▶ 821] The ADS reactive extensions are only available for usage, when the corresponding Nuget package is downloaded from Nuget.org. Beckhoff.TwinCAT.Ads.Reactive package on Nuget

Other Resources

HowTo Samples [▶ 1981]

Version History [▶ 14]

3.1 Prerequisites

Preconditions for installing the TwinCAT .NET ADS Communication API

- TwinCAT 2 or 3 Installation (XAR Runtime or full installation)
- Beckhoff.TwinCAT.Ads Nuget Package

  .NET Framework 4.0 or later
- Beckhoff.TwinCAT.Ads.Reactive Nuget Package

  .NET Framework 4.6 or later

3.2 Installation

The now preferred way to install the TwinCAT ADS .NET Communication API is to use the NuGet.org package manager. Alternatively, there is still the option to Add references manually via the "Add References..." Dialog in Visual Studio.

Beckhoff.TwinCAT.Ads package from Nuget.org repository.

This is the main package implementing the ADS client functionality. This is needed to establish ADS connections to local and remote devices.

Please follow the install instructions on the Nuget.org site.

Beckhoff.TwinCAT.Ads Nuget Package

Beckhoff.TwinCAT.Ads.Reactive package from Nuget.org repository.

This package installs Reactive extensions on top of the Beckhoff.TwinCAT.Ads Nuget Package and installs additional extensions to map ADSNotifications to observable events.

Please follow the install instructions on the Nuget.org site.

Beckhoff.TwinCAT.Ads.Reactive Nuget Package

Manually referencing the TwinCAT Ads .Net Communication Library (non-preferred, obsolete)

Add TwinCAT.Ads.dll manually in Visual Studio

1. Create New project: Start Microsoft Visual Studio and create new project (e.g. Windows Forms Application).
2. Adding reference: In order to select the TwinCAT.Ads class library you must choose the command Add Reference... under the Project menu . You will find the .Net Libraries per default in following TwinCAT folder:

   C:\TwinCAT\AdsApi\.NET\

   This opens the Add Reference dialog. In this dialog you have to press the Browse button and select the file TwinCAT.Ads.dll for your used .NET Framework runtime. In the Solution Explorer you can check, if the component has been added to the list of references.
3. All accessible types (classes,structures ...) belong to the namespace TwinCAT.Ads. Therefore one has to insert the following line at the beginning of the source:**C#**

   using System.IO; using TwinCAT.Ads;

This enables access to the types defined in TwinCAT.Ads without including the name of the namespace. The class TcAdsClient is the core of the TwinCAT.Ads class library and enables the user to communicate with an ads device. To begin with an instance of the class must be created. Then a connection to the ADS device is established by means of the Connect method.

4 Concepts

Concepts introduction

Concepts

Concepts discussed

Name	Description
IndexGroup / IndexOffset	Read/Write values by IndexGroup / IndexOffset [▶ 19]
Symbolic access	Read/Write values by symbolic instance path [▶ 22]
Access via symbol handle	Read/Write values by symbol handles. [▶ 23]
ADS Notifications	Concept of ADS Notifications [▶ 20]
Symbol loader access	Access symbolic information by Symbol loader [▶ 25]
Marshalling values via ANYTYPE concept	ANY Type blittable type marshalling [▶ 23]
Dynamic automatic marshalling via Dynamic Language Runtime	Use of the .NET Framework Dynamic Language Runtime (DLR) [▶ 26]
Access via reactive extensions	Observer value changes by ADS Reactive Extensions [▶ 27]

Other Resources

TwinCAT ADS .NET API Documentation [▶ 16]

4.1 Access Data via IndexGroup/IndexOffset

Reading/Writing values by Index/Group index offset are the most basic way to access data via ADS. This address combination directly link into the process image of virtual ADS Devices.

As long the process image is static this is unproblematic and a system near access, but if the content is more dynamic and the address changes over time the IndexGroup/IndexOffset can get invalid.

Examples about moving addresses could be:

- Changed Parametrization of IO (and Re-activation)
- The PLC Online change
- New Plc Downloads

In that case other access methods could be advantageous.

Another important point is that the data access is not type safe. The values are read or written to or from byte buffers and the proper marshalling/unmarshalling is the task of the application code.

Example

Access ProcessImage Data by IndexGroup/IndexOffset

```
using (TcAdsClient client = new TcAdsClient())
{
    UInt32 valueToRead = 0;
    UInt32 valueToWrite = 42;

    client.Connect(AmsNetId.Local, 851);

    // Write an UINT32 Value
    AdsStream writeStream = new AdsStream(sizeof(uint));
    AdsBinaryWriter writer = new AdsBinaryWriter(writeStream);
    writer.Write(valueToWrite);
```

```
    adsClient.Write(0x4020, 0x0, writeStream, 0, 4);

    // Read an UINT32 Value
    AdsStream readStream = new AdsStream(sizeof(uint));
    adsClient.Read(0x4020, 0x0, readStream, 0, 4);
    readStream.Position = 0;
    AdsBinaryReader reader = new AdsBinaryReader(readStream);
    valueToRead = reader.ReadUInt32();
}
```

4.2 Use of ADS Notifications

If values from a PLC or NC are to be displayed continuously on a user interface, then it is very inefficient to use asynchronous read access, since this function must be called cyclically (polled triggered by a timer). Instead of using a pulling (read) model, ADS Notifications are implementing a push model. That means ADS Notifications are triggered by the sender and form a single or series of ADS messages/events. Together with these ADS Notifications, values can be transmitted. A distinction is drawn between whether the TwinCAT server is to transmit the values cyclically, or only when the values change.

In principle (raw mode) a notification is begun with the registration of the notification AddDeviceNotification [▶ 657]. After this, events are automatically fired by TwinCAT. DeleteDeviceNotification [▶ 697] is used to halt the notification again. Since the number of notifications is limited, you should ensure the notifications no longer required by your program are unregistered/deleted.

There exist several 'modes' for different type of ADS Notification triggers. For a complete list please consult AdsTransMode [▶ 401].

All the following examples demonstrate how to receive ADS Notifications. The .NET ADS API supports different information layers which different levels of ADS Notification support. All are using a PLC variable in the PLC and each time the value of the PLC variable changes, an ADS Notification message is sent and the registered callback method is invoked with event arguments that contain all the necessary information (value, time stamp, ...).

Hint: Don't use time intensive executions or ADS commands inside of your callback (not more than approx. 500). Remind to sync your callback in your main thread (typically the UI thread) if necessary, because the ADS Notifications appear on a background thread.

Using ADS Notifications with Symbolic information

C#

```
private void SymbolValueChanged()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Connect to target
    client.Connect("1.2.3.4.5.6", 851);
    Symbol symbol = null;

    try
    {

        ISymbolLoader loader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
        // DINT Type (UINT32)
        symbol = (Symbol)loader.Symbols["MAIN.nCounter"];

        // Set the Notification Settings of the Symbol if NotificationSettings.Default is not approp
riate
        // Check for change every 500 ms
        symbol.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 500, 0);

        symbol.ValueChanged += Symbol_ValueChanged; // Registers the notification
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event and the underlying Handle
        symbol.ValueChanged -= Symbol_ValueChanged; // Unregisters the notification
    }
    }
```

```
}

private void Symbol_ValueChanged(object sender, ValueChangedArgs e)
{
    Symbol symbol = (Symbol)e.Symbol;

    // Object Value can be cast to int automatically, because it is an Primitive Value (DINT --
> Int32).
    // The Symbol information is used internally to cast the value to its appropriate .NET Type.
    int iVal = (int)e.Value;
}
```

Using ADS Notifications in 'ANYTYPE' style

C#

```
//AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event 'Ex' handler
    client.AdsNotificationEx += Client_AdsNotification;

    // Connect to target
    client.Connect(AmsNetId.Local, 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a ZDINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotificationEx("MAIN.nCounter",AdsTransMode.OnChange, 2
00, 0, null,typeof(uint));
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotificationEx -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationExEventArgs e)
{
    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = (uint)e.Value;
}
```

Using Raw ADS Notifications

C#

```
AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event handler
    client.AdsNotification += Client_AdsNotification;

    // Connect to target
    client.Connect("1.2.3.4.5.6", 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a DINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotification("MAIN.nCounter", readStream, AdsTransMode.
OnChange, 200, 0, null);
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
```

```
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotification -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationEventArgs e)
{
    int offset = (int)e.DataStream.Position;
    int length = (int)e.DataStream.Length;

    e.DataStream.Position = offset;
    AdsBinaryReader reader = new AdsBinaryReader(e.DataStream);

    // Read the Unmarshalled data
    //byte[] data = reader.ReadBytes(length);

    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = reader.ReadUInt32();
}
```

Using reactive ADS Notifications

C#

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o[1].CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    cycleCount.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 500, 5000); //
 optional: Change NotificationSettings on Symbol

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    IDisposable subscription = cycleCount.WhenValueChanged().Take(20).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

4.3 Access Data via Symbolic path

The Read/Write Access by symbol path solves the issue of directly accessing the process image. With specifying the access path to the symbol, the symbol address can be found by a binary search (internally) and reading / writing symbols is independent of the location within the process image.

This access method can only be used, when the ADS device is supporting symbolic information like the TwinCAT PLC.

Because its indirect access, the performance is slightly worse than the direct access via IndexGroup/ IndexOffset. However there are internal optimizations to cache handles to the already used symbols to accelerate repeated access.

Example

Access symbolic data by instance/symbol path

```
using (TcAdsClient client = new TcAdsClient())
{
    UInt32 valueToRead = 0;
    UInt32 valueToWrite = 42;

    client.Connect(AmsNetId.Local, 851);
    adsClient.WriteSymbol("MAIN.nCounter", valueToWrite, false);
    valueToRead = (uint)adsClient.ReadSymbol("MAIN.nCounter", typeof(UInt32),false);
}
```

4.4 Access Data via Symbol handles

The Read/Write Access by handle solves the issue of directly accessing the process image like the symbol path access. Because the address is accessed indirectly by the symbol path creating a variable handle, the read/write works also when the data object has changed its position within the process image.

However the cost for this are two extra ADS communication roundtrips by the 'CreateVariableHandle' and 'DeleteVariableHandle' calls compared to the IndexGroup/IndexOffset access methods. It is the responsibility of the application code to optimize these accesses.

Example

Access symbolic values by handle

```
using (TcAdsClient client = new TcAdsClient())
{
    int varHandle = 0;
    client.Connect(AmsNetId.Local, 851);
    try
    {
    UInt16 valueToRead = 0;
    UInt16 valueToWrite = 42;

    // Create the Variable Handle
    varHandle = client.CreateVariableHandle("MAIN.testVar"); //Test Var is defined as PLC INT

    // Write an UINT16 Value
    AdsStream writeStream = new AdsStream(sizeof(UInt16));
    AdsBinaryWriter writer = new AdsBinaryWriter(writeStream);
    writer.Write(valueToWrite); // Marshal the Value
    adsClient.Write(varHandle, writeStream,0,sizeof(uint));

    // Read an UINT16 Value
    AdsStream readStream = new AdsStream(sizeof(UInt16));
    adsClient.Read(varHandle, readStream, 0, sizeof(UInt16));
    readStream.Position = 0;
    AdsBinaryReader reader = new AdsBinaryReader(readStream);
    valueToRead = reader.ReadUInt16(); // Unmarshal the Value
    }
    finally
    {
    // Unregister VarHandle after Use
    client.DeleteVariableHandle(varHandle);
    }
}
```

4.5 Value marshalling with ANYTYPE concept

This topic describes reading and writing variables/symbols of 'any' type with the help of the ReadAny and WriteAny (ReadSymbol, WriteSymbol) methods. The value will be marshalled / cast directly from/to its appropriate .NET type, what eases the value access.

'Any' types in this context are all types that are 'blittable' to the process image - what means that the memory layout is equal (e.g primitive types) or is marshallable by the default marshalling of .NET (see 'PlcStruct' in the example below).

The appropriate .NET type must be known during compile time and is passed to the methods as parameter. In case of a ReadAny call, the read data will be returned as a object. The type of the object is marshalled to the type specified as parameter type. Because the data size and the memory alignment is taken from this type specification, it is so important that this specification fits to the memory representation in the ADS device (e.g. the PLC).

Because some data types(arrays and strings) need additional information, an overload of the method ReadAny exists, that takes an additional parameter args. A Full list of supported types can be found in the documentation of the overloaded method.

Reading and writing of structures

To be able to read or write PLC structures the memory layout of the structure or class in .NET must be the same as in the PLC. The layout of a structure or class can be specified with the attribute StructLayoutAttribute. The LayoutKind must be set to LayoutKind.Sequential and the pack must be set to 1

If arrays, strings or boolean values are define the class, one has to specify how these fields should be marshalled. This is accomplished with help of the MarshalAs attribute. Because arrays and strings do not have a fixed length in .NET, the property SizeConst is necessary for arrays and strings.

Marshalling values with 'ANY_TYPES'

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect("1.2.3.4.5.6", 851);

    // Bool value
    bool boolValue = (bool)client.ReadSymbol("MAIN.bool1", typeof(bool), false);
    adsClient.WriteSymbol("MAIN.bool1", boolValue,false);

    // or
    int handle1 = adsClient.CreateVariableHandle("MAIN.bool1"); // BOOL
    boolValue = (bool)client.ReadAny(handle1, typeof(bool));
    adsClient.WriteAny(handle1, boolValue);
    adsClient.DeleteVariableHandle(handle1);

    // RealValue
    int handle2 = adsClient.CreateVariableHandle("MAIN.bool1"); // BOOL
    float realValue = (float)client.ReadAny(handle2, typeof(float)); // REAL
    client.WriteAny(handle2, realValue);
    adsClient.DeleteVariableHandle(handle2);

    // String
    int handle3 = adsClient.CreateVariableHandle("MAIN.string1"); // STRING[80]
    string stringValue = (string)client.ReadAny(handle3, typeof(string), new int[] { 80 }); // Needs
 additional para for strlen
    adsClient.WriteAny(handle3, stringValue, new int[] { 80 });

    // ushort[]
    int handle4 = adsClient.CreateVariableHandle("MAIN.uint1Arr"); // ARRAY [0..9] OF UINT
    ushort[] ushortArr = (ushort[])client.ReadAny(handle4, typeof(ushort[]), new int[] { 10 });
    adsClient.WriteAny(handle4, ushortArr, new int[] { 10 });
    adsClient.DeleteVariableHandle(handle4);

    // Complex Struct Type
    // Take care the the corresponding .NET Type is blittable / marshallable to the PLC type
    int handle5 = adsClient.CreateVariableHandle("MAIN.struct");
    PlcStruct structValue = (PlcStruct)adsClient.ReadAny(handle5, typeof(PlcStruct));
    adsClient.WriteAny(handle5, typeof(PlcStruct));
    adsClient.DeleteVariableHandle(handle5);

    // ARRAY [0..9] OF STRING[80]
    // args[0] --> Number of Characters
    // args[1] --> Number of Array Elements
    // Needs additional para for strlen and number of Elements in Array
    int handle6 = adsClient.CreateVariableHandle("MAIN.stringArr"); // ARRAY [0..9] OF STRING[80]
    string[] stringArr = (string[])client.ReadAny(handle6, typeof(string[]), new int[] { 80, 10 });
    adsClient.WriteAny(handle6, stringValue, new int[] { 80 });
    adsClient.DeleteVariableHandle(handle6);

}
```

Defining Memory layout of struct type.

```
[StructLayout(LayoutKind.Sequential, Pack = 1, CharSet = CharSet.Ansi)]
public struct PlcStruct
{
    // Type must be 'blittable' to the corresponding PLC Struct Type
    // See MSDN for MarshalAs and Default Marshalling.

    [MarshalAs(UnmanagedType.I1)]
    public bool boolVal; // BOOL
    public byte byteVal; // BYTE
    public ushort ushortVal; // UINT
    public short shortVal; // INT
    public uint uintVal; // UDINT
    public int dintVal; // DINT
    public uint udintVal; // UDINT
    public float realVal; // REAL
    public double lrealVal; // LREAL
    [MarshalAs(UnmanagedType.ByValTStr, SizeConst = 81)]
    public string stringVal; // STRING[80]

    [MarshalAs(UnmanagedType.U4)]
    public uint timeVal; // TIME
    [MarshalAs(UnmanagedType.U4)]
    public uint todVal; // TOD
    [MarshalAs(UnmanagedType.U4)]
    public uint dateval; // DATE
    [MarshalAs(UnmanagedType.U4)]
    public uint dtVal; // DT
}
```

ADS Notifications with Type marshalling (AdsNotificationEx)

The method AddDeviceNotificationEx is used to register notifications for a PLC variable. If the value of a variable changes the event AdsNotificationEx is fired. The difference to the event AdsNotification, is that the value of the variable is stored in an object instead of in an AdsStream. Therefore one has to pass the type of the object to the method AddDeviceNotificationEx

Notifications with 'ANY_TYPES'

```
using (TcAdsClient client = new TcAdsClient())
{
    client.AdsNotificationEx += Client_AdsNotificationEx;
    client.Connect("1.2.3.4.5.6", 851);

    // Add UDINT
    int notificationHandle = client.AddDeviceNotificationEx("MAIN.udint", AdsTransMode.OnChange, 200
, 200, null, typeof(uint));
    Thread.Sleep(5000); // ...
    client.DeleteDeviceNotification(notificationHandle); // Unregister Event
}
```

Notifications with 'ANY_TYPES'

```
private void Client_AdsNotificationEx(object sender, AdsNotificationExEventArgs e)
{
    uint value = (uint)e.Value; // Marshalled value as .NET Type
}
```

4.6 Access Data via Symbol Loader

Some ADS Devices (e.g. the TwinCAT PLC) provide symbolic information for download. That means all visible Symbols and DataTypes can be retrieved from the target system. While this need an extra effort to upload and hold the data, this feature helps to remove the dependency of the code/configuration running on the target device.

E.g. because the symbolic information can now be browsed and determined during runtime, the application can be written without knowing what's running on the target system. Even more having the Symbol information cached, the access of the process image data will be easier because the datasize and access (instance path) is stored in the symbol.

Dependent how it is parametrized, the symbol loader can work with 'ANY_TYPES' (marshallable Primitive types, Value marshalling with ANYTYPE concept [▶ 23]) or full dynamic symbols (Automatic dynamic marshalling of values [▶ 26]).

Example

Accessing symbolic data by preloaded Symbolic information

```
using (TcAdsClient client = new TcAdsClient())
{
    UInt32 valueToRead = 0;
    UInt32 valueToWrite = 42;

    client.Connect(AmsNetId.Local, 851);

    // Load all Symbols + DataTypes
    ISymbolLoader loader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    Symbol symbol = (Symbol)loader.Symbols["MAIN.nCounter"];

    // Works for ALL Primitive 'ANY TYPES' Symbols
    symbol.WriteValue(valueToWrite);
    valueToRead = (UInt32)symbol.ReadValue();
}
```

4.7 Automatic dynamic marshalling of values

The 'Dynamic Symbol Loader' of the .NET ADS Communication API makes use of the .NET dynamic language runtime (DLR).

The dynamic language runtime is a runtime environment that adds a set of services for dynamic languages to the common language runtime (CLR). The DLR makes it easier to develop dynamic languages to run on the .NET Framework and to add dynamic features to statically typed languages.

Dynamic languages can identify the type of an object at run time, whereas in statically typed languages (without using the 'dynamic' keyword, specify object types at design time.

The advantage here is - from the moment on the symbolic (and dataType) information is available from the ADS Device - Symbol/Variable values can be marshalled 'on-the-fly' during runtime in a type-safe manner.

This works not only with primitive types but also with complex types. This reduces the complexity of the written application code to access the values, because neither the type of the data must be known, nor how the value data must be marshalled from/to the process image.

The price to be paid is simply that the full symbolic information and data types must be downloaded from the ADS Device by the symbol loader.

Example

Automatic marshalling values with 'Dynamic Values'

```
using (TcAdsClient client = new TcAdsClient())
{
    uint valueToRead = 0;
    uint valueToWrite = 42;

    client.Connect(AmsNetId.Local, 851);

    // Load all Symbols + DataTypes
    // Primitive Parts will be automatically resolved to .NET Primitive types.
    IDynamicSymbolLoader loader = (IDynamicSymbolLoader)SymbolLoaderFactory.Create(client, SymbolLoa
derSettings.DefaultDynamic);

    dynamic symbols = loader.SymbolsDynamic;
    dynamic main = symbols.Main;

    // Use typed object to use InfoTips
    DynamicSymbol nCounter = main.nCounter; // UDINT

    // or to be fullDynamic
```

```
    //dynamic nCounter = main.nCounter;

    // Works for ALL sorts of types (not restricted to ANY_TYPE basing primitive types)
    valueToRead = (uint)nCounter.ReadValue();
    // or
    var varValue = nCounter.ReadValue();
    // or
    dynamic dynValue = nCounter.ReadValue();

    // Same for writing
    nCounter.WriteValue(valueToWrite);

    // Or Notifications / Events
    nCounter.ValueChanged += NCounter_ValueChanged;

    //Reading complexTypes e.g. Struct

    DynamicSymbol myStructSymbol = main.plcStruct; // Dynamically created
    dynamic myStructVal = myStructSymbol.ReadValue(); // Takes an ADS Snapshot of the value

    dynamic int1Val = myStructVal.int1; // Value to an INT (short)
    dynamic valueNestedStruct = myStructVal.nestedStruct; //
value to another complex type (here a nested Struct)

    myStructSymbol.ValueChanged += MyStructSymbol_ValueChanged;
}
```

Automatic marshalling values with 'Dynamic Values'

```
private void NCounter_ValueChanged(object sender, ValueChangedArgs e)
{
    var uintVal = e.Value;
}

private void MyStructSymbol_ValueChanged(object sender, ValueChangedArgs e)
{
    dynamic structValue = e.Value; // Snapshot of the whole Struct and all its contents
}
```

Calling 'ReadValue' or the 'ValueChanged' notification takes a full snapshot (with snapshot time) of the value. That means, when for example subelements of a struct value will be accessed, all subvalues will represent the value of that snapshot time consistently. Starting point is always the 'DynamicSymbol' object that called 'ReadValue'.

An update of the value can be done directly on the value with 'UpdateValue', or with reading a new Value on the 'DynamicSymbol' ('ReadValue').

The 'ValueChanged' event on the 'DynamicSymbol' assigns a Notification for just this symbol. The 'ValueChanged' Handler will contain the value completely marshalled as dynamic object.

4.8 Reactive Extensions and ADS

The Reactive Extensions (Rx) is a .NET library for composing asynchronous and event-based programs using observable sequences and LINQ-style query operators. Using Rx, developers represent asynchronous data streams with Observables, query asynchronous data streams using LINQ operators, and parameterize the concurrency in the asynchronous data streams using Schedulers. Simply put, Rx = Observables + LINQ + Schedulers.

In ADS terms, not only the reading and writing data or symbol values can be put into reactive data streams, also ADS Notifications are a perfect fit for reactive code. This eases not only data binding to reactive frameworks (e.g. reactive UI) but also supports enhanced data manipulation via synchronous and asynchronous observers. Multithreaded and parallized code paths that support multiple CPU cores can be written very easily without the burden of deadlock and synchronization issues.

More about .NET reactive extensions can be read here: Reactive extensions project site.

The TwinCAT ADS Reactive extensions are available via a supplement Nuget Package: Beckhoff.TwinCAT.Ads.Reactive Nuget Package

Example

Observe for Notifications

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Reactive Notification Handler
    var valueObserver = Observer.Create<ushort>(val =>
    {
    Console.WriteLine(string.Format("Value: {0}", val.ToString()));
    }
    );

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    IDisposable subscription = client.WhenNotification<ushort>("TwinCAT_SystemInfoVarList._TaskInfo.
CycleCount", NotificationSettings.Default).Take(20).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Observe for Symbol Notifications

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o[1].CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    cycleCount.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 500, 5000); //
 optional: Change NotificationSettings on Symbol

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    IDisposable subscription = cycleCount.WhenValueChanged().Take(20).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Observer for dynamic Symbol Notifications

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = (IDynamicSymbolLoader)SymbolLoaderFactory.Create(client, SymbolLoaderSettings
.DefaultDynamic);
    dynamic symbols = symbolLoader.SymbolsDynamic;
    dynamic cycleCount = symbols.TwinCAT_SystemInfoVarList._TaskInfo[1].CycleCount;

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    // Value objects can be dynamically (on the fly) created objects here (e.g. structs)
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
```

```
    }
    );

    cycleCount.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 500, 5000); //
 optional: Change NotificationSettings on Symbol

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    // We have to give the 'hint' about IValueSymbol here, that the CLR finds the Extension Method '
WhenValueChanged' during runtime.
    IDisposable subscription = ((IValueSymbol)cycleCount).WhenValueChanged().Take(20).Subscribe(valu
eObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Polling observer

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o.CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    // Take 20 Values in an Interval of 500ms
    IDisposable subscription = cycleCount.PollValues(TimeSpan.FromMilliseconds(500)).Take(20).Subscr
ibe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Writing values with observable subject

```
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information (Symbol 'i : INT' in PLC Global Variables list.
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol gvlIntSymbol = (IValueSymbol)symbolLoader.Symbols["GVL.i"];

    // Produces object (short) Values 0,1,2,3 ... in seconds period
    IObservable<object> timerObservable = Observable.Interval(TimeSpan.FromSeconds(1.0)).Select(i =>
 (object)(short)i);

    // Take 10 Values (0..9) and write them to GVL.i
    IDisposable dispose = gvlIntSymbol.WriteValues(timerObservable.Take(10));

    Console.ReadKey(); // Wait for Key press
    dispose.Dispose(); // Dispose the Subscription
}
```

5 TwinCAT.Ads Namespaces

TwinCAT ADS .NET API

Namespaces

Namespace	Description
TwinCAT [▶ 30]	Common namespace for types that are not specific to ADS.
TwinCAT.Ads [▶ 108]	ADS root namespace.
TwinCAT.Ads.Reactive [▶ 820]	Reactive Extensions for the ADS Client. All types within are contained in the ADS companion package "Beckhoff.TwinCAT.Ads.Reactive" which must be referenced seperately. https://www.nuget.org/packages/Beckhoff.TwinCAT.Ads.Reactive/
TwinCAT.Ads.SumCommand [▶ 892]	ADS offers powerful and fast communication to exchange any kind of information. It's possible to read single variables or complete arrays and structures with each one single ADS-API call. ADS Sum-Commands offer to read/write with one single ADS call multiple variables which are not structured within a linear memory, effectivly reducing roundtrips.
TwinCAT.Ads.TypeSystem [▶ 923]	Root namespace for the ADS type system.
TwinCAT.Ads.ValueAccess [▶ 1119]	Root namespace for ADS value access.
TwinCAT.PlcOpen [▶ 1120]	
TwinCAT.TypeSystem [▶ 1184]	Namespace for the common (non ADS dependant) type system.
TwinCAT.TypeSystem.Generic [▶ 1854]	Namespace for the dynamic part of the common type system.
TwinCAT.ValueAccess [▶ 1978]	Namespace for the common (non ADS dependant) value access.

5.1 TwinCAT Namespace

Common namespace for types that are not specific to ADS.

Classes

	Class	Description
	CannotAccessVirtualSymbolException [▶ 31]	Cannot access virtual Symbol
	ClientNotConnectedException [▶ 36]	Class ClientNotConnectedException
	ConnectionStateChangedEventArgs [▶ 39]	Event arguments for the Connection status changed event.
	InsufficientAccessRights [▶ 56]	Insufficient rights for access

	Class	Description
	Session [▶ 67]	Abstract Session base class.
	SessionConnectionStateChangedEventArgs [▶ 84]	EventArguments for the ConnectionStatusChanged events.
	SessionException [▶ 90]	Session Exception
	SessionNotConnectedException [▶ 97]	Class SessionNotConnectedException.
	SessionProvider.S, A, C. [▶ 102]	Abstract base class for a Custom Session provider

Interfaces

	Interface	Description
	IConnection [▶ 46]	Interface IConnection
	IConnectionStateProvider [▶ 52]	Interface IConnectionStateProvider
	ISession [▶ 60]	Interface ISession
	ISessionSettings [▶ 66]	Interface ISessionSettings
	ISymbolLoaderSettings [▶ 66]	Interface ISymbolLoaderSettings

Enumerations

	Enumeration	Description
	ConnectionState [▶ 39]	Connection state enumeration
	ConnectionStateChangedReason [▶ 46]	Reason for the Connection status changed event.
	SessionProviderCapabilities [▶ 107]	Enum SessionProviderCapabilities
	SymbolsLoadMode [▶ 108]	Enum SymbolsLoadMode

5.1.1 CannotAccessVirtualSymbolException Class

Cannot access virtual Symbol

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.Ads.SymbolException [▶ 606]
          TwinCAT.CannotAccessVirtualSymbolException

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class CannotAccessVirtualSymbolException : SymbolException
```

VB

```
<SerializableAttribute>
Public Class CannotAccessVirtualSymbolException
    Inherits SymbolException
```

The CannotAccessVirtualSymbolException type exposes the following members.

Constructors

	Name	Description
	CannotAccessVirtualSymbolException [▶ 33]	Initializes a new instance of the CannotAccessVirtualSymbolException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Fields

	Name	Description
	Symbol [▶ 615]	Symbol that is bound to the SymbolException [▶ 606] (Inherited from SymbolException [▶ 606].)

Reference

TwinCAT Namespace [▶ 30]

5.1.1.1 CannotAccessVirtualSymbolException Constructor

Initializes a new instance of the CannotAccessVirtualSymbolException [▶ 31] class.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public CannotAccessVirtualSymbolException(
    ISymbol symbol
)
```

VB

```
Public Sub New (
    symbol As ISymbol
)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The symbol.

Reference

CannotAccessVirtualSymbolException Class [▶ 31]

TwinCAT Namespace [▶ 30]

5.1.1.2 CannotAccessVirtualSymbolException Properties

The CannotAccessVirtualSymbolException [▶ 31] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

CannotAccessVirtualSymbolException Class [▶ 31]

TwinCAT Namespace [▶ 30]

5.1.1.3 CannotAccessVirtualSymbolException Methods

The CannotAccessVirtualSymbolException [▶ 31] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

CannotAccessVirtualSymbolException Class [▶ 31]

TwinCAT Namespace [▶ 30]

5.1.1.4 CannotAccessVirtualSymbolException Events

The CannotAccessVirtualSymbolException [▶ 31] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

CannotAccessVirtualSymbolException Class [▶ 31]

TwinCAT Namespace [▶ 30]

5.1.1.5 CannotAccessVirtualSymbolException Fields

The CannotAccessVirtualSymbolException [▶ 31] type exposes the following members.

Fields

	Name	Description
	Symbol [▶ 615]	Symbol that is bound to the SymbolException [▶ 606] (Inherited from SymbolException [▶ 606].)

Reference

CannotAccessVirtualSymbolException Class [▶ 31]

TwinCAT Namespace [▶ 30]

5.1.2 ClientNotConnectedException Class

Class ClientNotConnectedException

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.ClientNotConnectedException

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class ClientNotConnectedException : AdsException
```

VB

```
<SerializableAttribute>
Public Class ClientNotConnectedException
    Inherits AdsException
```

The ClientNotConnectedException type exposes the following members.

Constructors

	Name	Description
	ClientNotConnectedException [▶ 37]	Initializes a new instance of the ClientNotConnectedException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

TwinCAT Namespace [▶ 30]

5.1.2.1 ClientNotConnectedException Constructor

Initializes a new instance of the ClientNotConnectedException [▶ 36] class.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ClientNotConnectedException()
```

VB

```
Public Sub New
```

Reference

ClientNotConnectedException Class [▶ 36]

TwinCAT Namespace [▶ 30]

5.1.2.2 ClientNotConnectedException Properties

The ClientNotConnectedException [▶ 36] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

ClientNotConnectedException Class [▶ 36]

TwinCAT Namespace [▶ 30]

5.1.2.3 ClientNotConnectedException Methods

The ClientNotConnectedException [▶ 36] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

ClientNotConnectedException Class [▶ 36]

TwinCAT Namespace [▶ 30]

5.1.2.4 ClientNotConnectedException Events

The ClientNotConnectedException [▶ 36] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

ClientNotConnectedException Class [▶ 36]

TwinCAT Namespace [▶ 30]

5.1.3 ConnectionState Enumeration

Connection state enumeration

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public enum ConnectionState
```

VB

```
Public Enumeration ConnectionState
```

Members

	Member name	Value	Description
	None	0	Unknown / Uninitialized
	Unknown	0	Unknown / Uninitialized
	Disconnected	1	Disconnected
	Connected	2	Connected
	Lost	3	Connection lost

Reference

TwinCAT Namespace [▶ 30]

5.1.4 ConnectionStateChangedEventArgs Class

Event arguments for the Connection status changed event.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.ConnectionStateChangedEventArgs
      TwinCAT.SessionConnectionStateChangedEventArgs [▶ 84]

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ConnectionStateChangedEventArgs : EventArgs
```

VB

```
Public Class ConnectionStateChangedEventArgs
    Inherits EventArgs
```

The ConnectionStateChangedEventArgs type exposes the following members.

Constructors

	Name	Description
	ConnectionStateChangedEventArgs(ConnectionStateChangedReason, ConnectionState, ConnectionState) [▶ 41]	Constructs the ConnectionStateChangedEventArgs arguments.
	ConnectionStateChangedEventArgs(ConnectionStateChangedReason, ConnectionState, ConnectionState, Exception) [▶ 42]	Constructs the ConnectionStateChangedEventArgs arguments.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Fields

	Name	Description
	Exception [▶ 44]	Exception, (only for Error [▶ 46]
	NewState [▶ 44]	New connection state
	OldState [▶ 45]	Old connection state
	Reason [▶ 45]	Reason for the event

Reference

TwinCAT Namespace [▶ 30]

5.1.4.1 ConnectionStateChangedEventArgs Constructor

Overload List

	Name	Description
	ConnectionStateChangedEventArgs(ConnectionStateChangedReason, ConnectionState, ConnectionState) [▶ 41]	Constructs the ConnectionStateChangedEventArgs [▶ 39] arguments.
	ConnectionStateChangedEventArgs(ConnectionStateChangedReason, ConnectionState, ConnectionState, Exception) [▶ 42]	Constructs the ConnectionStateChangedEventArgs [▶ 39] arguments.

Reference

ConnectionStateChangedEventArgs Class [▶ 39]

TwinCAT Namespace [▶ 30]

5.1.4.1.1 ConnectionStateChangedEventArgs Constructor (ConnectionStateChangedReason, ConnectionState, ConnectionState)

Constructs the ConnectionStateChangedEventArgs [▶ 39] arguments.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ConnectionStateChangedEventArgs(
    ConnectionStateChangedReason reason,
    ConnectionState newState,
    ConnectionState oldState
)
```

VB

```
Public Sub New (
    reason As ConnectionStateChangedReason,
    newState As ConnectionState,
    oldState As ConnectionState
)
```

Parameters

reason	Type: TwinCAT.ConnectionStateChangedReason [▶ 46] The reason.
newState	Type: TwinCAT.ConnectionState [▶ 39] The new state.
oldState	Type: TwinCAT.ConnectionState [▶ 39] The old state.

Reference

ConnectionStateChangedEventArgs Class [▶ 39]

ConnectionStateChangedEventArgs Overload [▶ 41]

TwinCAT Namespace [▶ 30]

5.1.4.1.2 ConnectionStateChangedEventArgs Constructor (ConnectionStateChangedReason, ConnectionState, ConnectionState, Exception)

Constructs the ConnectionStateChangedEventArgs [▶ 39] arguments.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ConnectionStateChangedEventArgs(
    ConnectionStateChangedReason reason,
    ConnectionState newState,
    ConnectionState oldState,
    Exception e
)
```

VB

```
Public Sub New (
    reason As ConnectionStateChangedReason,
    newState As ConnectionState,
    oldState As ConnectionState,
    e As Exception
)
```

Parameters

reason	Type: TwinCAT.ConnectionStateChangedReason [▶ 46] The reason.
newState	Type: TwinCAT.ConnectionState [▶ 39] The new state.
oldState	Type: TwinCAT.ConnectionState [▶ 39] The old state.
e	Type: System.Exception The e.

Reference

ConnectionStateChangedEventArgs Class [▶ 39]

ConnectionStateChangedEventArgs Overload [▶ 41]

TwinCAT Namespace [▶ 30]

5.1.4.2 ConnectionStateChangedEventArgs Methods

The ConnectionStateChangedEventArgs [▶ 39] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

ConnectionStateChangedEventArgs Class [▶ 39]

TwinCAT Namespace [▶ 30]

5.1.4.3 ConnectionStateChangedEventArgs Fields

The ConnectionStateChangedEventArgs [▶ 39] type exposes the following members.

Fields

	Name	Description
	Exception [▶ 44]	Exception, (only for Error [▶ 46]
	NewState [▶ 44]	New connection state

	Name	Description
	OldState [▶ 45]	Old connection state
	Reason [▶ 45]	Reason for the event

Reference

ConnectionStateChangedEventArgs Class [▶ 39]

TwinCAT Namespace [▶ 30]

5.1.4.3.1 ConnectionStateChangedEventArgs.Exception Field

Exception, (only for Error [▶ 46]

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly Exception Exception
```

VB

```
Public ReadOnly Exception As Exception
```

Field Value

Type: Exception

Reference

ConnectionStateChangedEventArgs Class [▶ 39]

TwinCAT Namespace [▶ 30]

5.1.4.3.2 ConnectionStateChangedEventArgs.NewState Field

New connection state

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly ConnectionState NewState
```

VB

```
Public ReadOnly NewState As ConnectionState
```

Field Value

Type: ConnectionState [▶ 39]

Reference

ConnectionStateChangedEventArgs Class [▶ 39]

TwinCAT Namespace [▶ 30]

5.1.4.3.3 ConnectionStateChangedEventArgs.OldState Field

Old connection state

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly ConnectionState OldState
```

VB

```
Public ReadOnly OldState As ConnectionState
```

Field Value

Type: ConnectionState [▶ 39]

Reference

ConnectionStateChangedEventArgs Class [▶ 39]

TwinCAT Namespace [▶ 30]

5.1.4.3.4 ConnectionStateChangedEventArgs.Reason Field

Reason for the event

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly ConnectionStateChangedReason Reason
```

VB

```
Public ReadOnly Reason As ConnectionStateChangedReason
```

Field Value

Type: ConnectionStateChangedReason [▶ 46]

Reference

ConnectionStateChangedEventArgs Class [▶ 39]

TwinCAT Namespace [▶ 30]

5.1.5 ConnectionStateChangedReason Enumeration

Reason for the Connection status changed event.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public enum ConnectionStateChangedReason
```

VB

```
Public Enumeration ConnectionStateChangedReason
```

Members

	Member name	Value	Description
	None	0	None / Uninitialized
	Established	1	The connection to the target has been established
	Closed	2	The Connection was closed
	Lost	3	The connection to the target has been lost
	Error	4	Communication error to the target (the connection is not shutting down)
	Resurrected	5	Communication was resurrected (available again)

Reference

TwinCAT Namespace [▶ 30]

5.1.6 IConnection Interface

Interface IConnection

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IConnection : IConnectionStateProvider
```

VB

```
Public Interface IConnection
    Inherits IConnectionStateProvider
```

The IConnection type exposes the following members.

Properties

	Name	Description
	ConnectionState [▶ 53]	Gets the current Connection state of the IConnectionStateProvider [▶ 52] (Inherited from IConnectionStateProvider [▶ 52].)

	Name	Description
	Id [▶ 48]	Gets the Connection Identifier .
	IsConnected [▶ 48]	Gets a value indicating whether the local ADS port was opened successfully. It does not indicate if the target port is available. Use the method ReadState to determine if the target port is available.
	Session [▶ 49]	Gets the session that initiated this IConnection
	Timeout [▶ 49]	Gets the timeout (in milliseconds)

Methods

	Name	Description
	Close [▶ 50]	Closes this IConnection
	Connect [▶ 50]	(Re)Connects the IConnection when disconnected.
	Disconnect [▶ 51]	Disconnects this IConnection.

Events

	Name	Description
	ConnectionStateChanged [▶ 54]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed. (Inherited from IConnectionStateProvider [▶ 52].)

Reference

TwinCAT Namespace [▶ 30]

5.1.6.1 IConnection Properties

The IConnection [▶ 46] type exposes the following members.

Properties

	Name	Description
	ConnectionState [▶ 53]	Gets the current Connection state of the IConnectionStateProvider [▶ 52] (Inherited from IConnectionStateProvider [▶ 52].)
	Id [▶ 48]	Gets the Connection Identifier .
	IsConnected [▶ 48]	Gets a value indicating whether the local ADS port was opened successfully. It does not indicate if the target port is available. Use the method ReadState to determine if the target port is available.
	Session [▶ 49]	Gets the session that initiated this IConnection [▶ 46]
	Timeout [▶ 49]	Gets the timeout (in milliseconds)

Reference

IConnection Interface [▶ 46]

TwinCAT Namespace [▶ 30]

5.1.6.1.1 IConnection.Id Property

Gets the Connection Identifier .

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Id { get; }
```

VB

```
ReadOnly Property Id As Integer
    Get
```

Property Value

Type: Int32
The identifier.

Reference

IConnection Interface [▶ 46]

TwinCAT Namespace [▶ 30]

5.1.6.1.2 IConnection.IsConnected Property

Gets a value indicating whether the local ADS port was opened successfully. It does not indicate if the target port is available. Use the method ReadState to determine if the target port is available.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsConnected { get; }
```

VB

```
ReadOnly Property IsConnected As Boolean
    Get
```

Property Value

Type: Boolean

Reference

IConnection Interface [▶ 46]

TwinCAT Namespace [▶ 30]

5.1.6.1.3 IConnection.Session Property

Gets the session that initiated this IConnection [▶ 46]

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ISession Session { get; }
```

VB

```
ReadOnly Property Session As ISession
    Get
```

Property Value

Type: ISession [▶ 60]
The session or NULL

Remarks

The Session can be null on standalone connections.

Reference

IConnection Interface [▶ 46]

TwinCAT Namespace [▶ 30]

5.1.6.1.4 IConnection.Timeout Property

Gets the timeout (in milliseconds)

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Timeout { get; set; }
```

VB

```
Property Timeout As Integer
    Get
    Set
```

Property Value

Type: Int32
The timeout.

Reference

IConnection Interface [▶ 46]

TwinCAT Namespace [▶ 30]

5.1.6.2 IConnection Methods

The IConnection [▶ 46] type exposes the following members.

Methods

	Name	Description
	Close [▶ 50]	Closes this IConnection [▶ 46]
	Connect [▶ 50]	(Re)Connects the IConnection [▶ 46] when disconnected.
	Disconnect [▶ 51]	Disconnects this IConnection [▶ 46].

Reference

IConnection Interface [▶ 46]

TwinCAT Namespace [▶ 30]

5.1.6.2.1 IConnection.Close Method

Closes this IConnection [▶ 46]

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void Close()
```

VB

```
Sub Close
```

Remarks

Closed Connections cannot be reconnected (when disposable objects behind)

Reference

IConnection Interface [▶ 46]

TwinCAT Namespace [▶ 30]

5.1.6.2.2 IConnection.Connect Method

(Re)Connects the IConnection [▶ 46] when disconnected.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool Connect()
```

VB

```
Function Connect As Boolean
```

Return Value

Type: Boolean

Reference

IConnection Interface [▶ 46]

TwinCAT Namespace [▶ 30]

5.1.6.2.3 IConnection.Disconnect Method

Disconnects this IConnection [▶ 46].

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool Disconnect()
```

VB

```
Function Disconnect As Boolean
```

Return Value

Type: Boolean

Reference

IConnection Interface [▶ 46]

TwinCAT Namespace [▶ 30]

5.1.6.3 IConnection Events

The IConnection [▶ 46] type exposes the following members.

Events

	Name	Description
	ConnectionStateChanged [▶ 54]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed. (Inherited from IConnectionStateProvider [▶ 52].)

Reference

IConnection Interface [▶ 46]

TwinCAT Namespace [▶ 30]

5.1.7 IConnectionStateProvider Interface

Interface IConnectionStateProvider

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IConnectionStateProvider
```

VB

```
Public Interface IConnectionStateProvider
```

The IConnectionStateProvider type exposes the following members.

Properties

	Name	Description
	ConnectionState [▶ 53]	Gets the current Connection state of the IConnectionStateProvider

Events

	Name	Description
	ConnectionStateChanged [▶ 54]	Occurs when connection status of the IConnectionStateProvider has been changed.

Reference

TwinCAT Namespace [▶ 30]

5.1.7.1 IConnectionStateProvider Properties

The IConnectionStateProvider [▶ 52] type exposes the following members.

Properties

	Name	Description
	ConnectionState [▶ 53]	Gets the current Connection state of the IConnectionStateProvider [▶ 52]

Reference

IConnectionStateProvider Interface [▶ 52]

TwinCAT Namespace [▶ 30]

5.1.7.1.1 IConnectionStateProvider.ConnectionState Property

Gets the current Connection state of the IConnectionStateProvider [▶ 52]

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ConnectionState ConnectionState { get; }
```

VB

```
ReadOnly Property ConnectionState As ConnectionState
    Get
```

Property Value

Type: ConnectionState [▶ 39]
The state of the connection.

Remarks

The Connection state changes only if the IConnection [▶ 46] is established / shut down or active communication is triggered by the User of the IConnection [▶ 46] object.

Examples

The following sample shows how to keep the ConnectionState updated by triggering ADS Communication.

Trigger ConnectionState changes in WPF Applications

```
private DispatcherTimer _timer = null;
private AdsSession _session = null;
//private AdsConnection _connection = null;

private void Window_Loaded(object sender, RoutedEventArgs e)
{
    _session = new AdsSession(AmsNetId.Local, 10000);
    IConnection connection = _session.Connect();
    tbConnectionState.Text = connection.ConnectionState.ToString();
    _session.ConnectionStateChanged += _session_ConnectionStateChanged;

    _timer = new DispatcherTimer();
    _timer.Interval = TimeSpan.FromMilliseconds(200);
    _timer.Tick += TimerOnTick;

    _timer.Start();
}

private void Window_Unloaded(object sender, RoutedEventArgs e)
{
    _timer.Stop();
    _session.Dispose();
}

private void _session_ConnectionStateChanged(object sender, TwinCAT.ConnectionStateChangedEventArgs
e)
{
    // ConnectionStateChanged will be triggered by communication Invokes
```

```
    tbConnectionState.Text = e.NewState.ToString();
}

private void TimerOnTick(object sender, EventArgs eventArgs)
{
    // The Timer Event will occur here in the UI thread because its an DispatcherTimer event!
    // An active ADS request will trigger Connection State periodically!
    StateInfo stateInfo;
    if (_session.Connection.TryReadState(out stateInfo) == AdsErrorCode.NoError)
    {
    tbAdsState.Text = stateInfo.AdsState.ToString();
    }
    else
    {
    tbAdsState.Text = "Invalid";
    }
}
```

Reference

IConnectionStateProvider Interface [▶ 52]

TwinCAT Namespace [▶ 30]

IConnectionStateProvider.ConnectionStateChanged [▶ 54]

5.1.7.2 IConnectionStateProvider Events

The IConnectionStateProvider [▶ 52] type exposes the following members.

Events

	Name	Description
	ConnectionStateChanged [▶ 54]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed.

Reference

IConnectionStateProvider Interface [▶ 52]

TwinCAT Namespace [▶ 30]

5.1.7.2.1 IConnectionStateProvider.ConnectionStateChanged Event

Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
event EventHandler<ConnectionStateChangedEventArgs> ConnectionStateChanged
```

VB

```
Event ConnectionStateChanged As EventHandler(Of ConnectionStateChangedEventArgs)
```

Value

Type: System.EventHandler.ConnectionStateChangedEventArgs [▶ 39].

Remarks

The Connection state changes only if the IConnection [▶ 46] is established / shut down or active communication is triggered by the User of the IConnection [▶ 46] object.

Examples

The following sample shows how to keep the ConnectionState [▶ 53] updated by triggering ADS Communication.

Trigger ConnectionState changes in WPF Applications

```
private DispatcherTimer _timer = null;
private AdsSession _session = null;
//private AdsConnection _connection = null;

private void Window_Loaded(object sender, RoutedEventArgs e)
{
    _session = new AdsSession(AmsNetId.Local, 10000);
    IConnection connection = _session.Connect();
    tbConnectionState.Text = connection.ConnectionState.ToString();
    _session.ConnectionStateChanged += _session_ConnectionStateChanged;

    _timer = new DispatcherTimer();
    _timer.Interval = TimeSpan.FromMilliseconds(200);
    _timer.Tick += TimerOnTick;

    _timer.Start();
}

private void Window_Unloaded(object sender, RoutedEventArgs e)
{
    _timer.Stop();
    _session.Dispose();
}

private void _session_ConnectionStateChanged(object sender, TwinCAT.ConnectionStateChangedEventArgs
e)
{
    // ConnectionStateChanged will be triggered by communication Invokes
    tbConnectionState.Text = e.NewState.ToString();
}

private void TimerOnTick(object sender, EventArgs eventArgs)
{
    // The Timer Event will occur here in the UI thread because its an DispatcherTimer event!
    // An active ADS request will trigger Connection State periodically!
    StateInfo stateInfo;
    if (_session.Connection.TryReadState(out stateInfo) == AdsErrorCode.NoError)
    {
    tbAdsState.Text = stateInfo.AdsState.ToString();
    }
    else
    {
    tbAdsState.Text = "Invalid";
    }
}
```

Reference

IConnectionStateProvider Interface [▶ 52]

TwinCAT Namespace [▶ 30]

IConnectionStateProvider.ConnectionState [▶ 53]

5.1.8 InsufficientAccessRights Class

Insufficient rights for access

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.Ads.SymbolException [▶ 606]
          TwinCAT.InsufficientAccessRights

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class InsufficientAccessRights : SymbolException
```

VB

```
<SerializableAttribute>
Public Class InsufficientAccessRights
    Inherits SymbolException
```

The InsufficientAccessRights type exposes the following members.

Constructors

	Name	Description
	InsufficientAccessRights [▶ 57]	Initializes a new instance of the InsufficientAccessRights class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Fields

	Name	Description
	Symbol [▶ 615]	Symbol that is bound to the SymbolException [▶ 606] (Inherited from SymbolException [▶ 606].)

Reference

TwinCAT Namespace [▶ 30]

5.1.8.1 InsufficientAccessRights Constructor

Initializes a new instance of the InsufficientAccessRights [▶ 56] class.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public InsufficientAccessRights(
    IValueSymbol symbol,
    SymbolAccessRights requested
)
```

VB

```
Public Sub New (
    symbol As IValueSymbol,
    requested As SymbolAccessRights
)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.IValueSymbol [▶ 1683] The symbol.
requested	Type: TwinCAT.TypeSystem.SymbolAccessRights [▶ 1818] The requested.

Reference

InsufficientAccessRights Class [▶ 56]

TwinCAT Namespace [▶ 30]

5.1.8.2 InsufficientAccessRights Properties

The InsufficientAccessRights [▶ 56] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

InsufficientAccessRights Class [▶ 56]

TwinCAT Namespace [▶ 30]

5.1.8.3 InsufficientAccessRights Methods

The InsufficientAccessRights [▶ 56] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

InsufficientAccessRights Class [▶ 56]

TwinCAT Namespace [▶ 30]

5.1.8.4 InsufficientAccessRights Events

The InsufficientAccessRights [▶ 56] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

InsufficientAccessRights Class [▶ 56]

TwinCAT Namespace [▶ 30]

5.1.8.5 InsufficientAccessRights Fields

The InsufficientAccessRights [▶ 56] type exposes the following members.

Fields

	Name	Description
	Symbol [▶ 615]	Symbol that is bound to the SymbolException [▶ 606] (Inherited from SymbolException [▶ 606].)

Reference

InsufficientAccessRights Class [▶ 56]

TwinCAT Namespace [▶ 30]

5.1.9 ISession Interface

Interface ISession

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISession : IConnectionStateProvider
```

VB

```
Public Interface ISession
    Inherits IConnectionStateProvider
```

The ISession type exposes the following members.

Properties

	Name	Description
	AddressSpecifier [▶ 61]	Gets the communication endpoint address string representation.
	Connection [▶ 62]	Gets the Connection object.
	ConnectionState [▶ 53]	Gets the current Connection state of the IConnectionStateProvider [▶ 52] (Inherited from IConnectionStateProvider [▶ 52].)
	EstablishedAt [▶ 62]	Gets the UTC time when the session was established.
	Id [▶ 63]	Gets the Session Id
	IsConnected [▶ 63]	Gets a value indicating whether the session is connected.

Methods

	Name	Description
	Close [▶ 64]	Closes this ISession
	Connect [▶ 65]	Connects the session and returns the established IConnection [▶ 46] object.
	Disconnect [▶ 65]	Disconnects the ISession

Events

	Name	Description
	ConnectionStateChanged [▶ 54]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed. (Inherited from IConnectionStateProvider [▶ 52].)

Reference

TwinCAT Namespace [▶ 30]

5.1.9.1 ISession Properties

The ISession [▶ 60] type exposes the following members.

Properties

	Name	Description
	AddressSpecifier [▶ 61]	Gets the communication endpoint address string representation.
	Connection [▶ 62]	Gets the Connection object.
	ConnectionState [▶ 53]	Gets the current Connection state of the IConnectionStateProvider [▶ 52] (Inherited from IConnectionStateProvider [▶ 52].)
	EstablishedAt [▶ 62]	Gets the UTC time when the session was established.
	Id [▶ 63]	Gets the Session Id
	IsConnected [▶ 63]	Gets a value indicating whether the session is connected.

Reference

ISession Interface [▶ 60]

TwinCAT Namespace [▶ 30]

5.1.9.1.1 ISession.AddressSpecifier Property

Gets the communication endpoint address string representation.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string AddressSpecifier { get; }
```

VB

```
ReadOnly Property AddressSpecifier As String
    Get
```

Property Value

Type: String
The address.

Reference

ISession Interface [▶ 60]

TwinCAT Namespace [▶ 30]

5.1.9.1.2 ISession.Connection Property

Gets the Connection object.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IConnection Connection { get; }
```

VB

```
ReadOnly Property Connection As IConnection
    Get
```

Property Value

Type: IConnection [▶ 46]
The connection.

Remarks

The IConnection [▶ 46] object is established by the ISession [▶ 60] via Connect. [▶ 65] and is valid until the Disconnect. [▶ 65] method or the Dispose method is called. Any possible resurrections after communication losses will be done transparently under the hood of the IConnection [▶ 46] so that the IConnection [▶ 46] instance and ISession [▶ 60] instance.

Reference

ISession Interface [▶ 60]

TwinCAT Namespace [▶ 30]

5.1.9.1.3 ISession.EstablishedAt Property

Gets the UTC time when the session was established.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
DateTime EstablishedAt { get; }
```

VB

```
ReadOnly Property EstablishedAt As DateTime
    Get
```

Property Value

Type: DateTime
The session established at.

Reference

ISession Interface [▶ 60]

TwinCAT Namespace [▶ 30]

5.1.9.1.4 ISession.Id Property

Gets the Session Id

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Id { get; }
```

VB

```
ReadOnly Property Id As Integer
    Get
```

Property Value

Type: Int32
The identifier.

Reference

ISession Interface [▶ 60]

TwinCAT Namespace [▶ 30]

5.1.9.1.5 ISession.IsConnected Property

Gets a value indicating whether the session is connected.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsConnected { get; }
```

VB

```
ReadOnly Property IsConnected As Boolean
    Get
```

Property Value

Type: Boolean
true if session is connected; otherwise, false.

Reference

ISession Interface [▶ 60]

TwinCAT Namespace [▶ 30]

5.1.9.2 ISession Methods

The ISession [▶ 60] type exposes the following members.

Methods

	Name	Description
	Close [▶ 64]	Closes this ISession [▶ 60]
	Connect [▶ 65]	Connects the session and returns the established IConnection [▶ 46] object.
	Disconnect [▶ 65]	Disconnects the ISession [▶ 60]

Reference

ISession Interface [▶ 60]

TwinCAT Namespace [▶ 30]

5.1.9.2.1 ISession.Close Method

Closes this ISession [▶ 60]

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void Close()
```

VB

```
Sub Close
```

Remarks

Closes also the IConnection [▶ 46].

Reference

ISession Interface [▶ 60]

TwinCAT Namespace [▶ 30]

5.1.9.2.2 ISession.Connect Method

Connects the session and returns the established IConnection [▶ 46] object.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IConnection Connect()
```

VB

```
Function Connect As IConnection
```

Return Value

Type: IConnection [▶ 46]
The IConnection [▶ 46] object.

Remarks

The IConnection [▶ 46] will be valid until the ISession [▶ 60] is disconnected via the Disconnect. [▶ 65] method or the Dispose method is called. Any possible resurrections after communication losses will be done transparently within the IConnection [▶ 46] so that the IConnection [▶ 46] instance and ISession [▶ 60] instance remains.

Reference

ISession Interface [▶ 60]

TwinCAT Namespace [▶ 30]

5.1.9.2.3 ISession.Disconnect Method

Disconnects the ISession [▶ 60]

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool Disconnect()
```

VB

```
Function Disconnect As Boolean
```

Return Value

Type: Boolean
true if Session was disconnected, false if the session was already closed.

Remarks

Disposes also the IConnection [▶ 46].

Reference

ISession Interface [▶ 60]

TwinCAT Namespace [▶ 30]

5.1.9.3 ISession Events

The ISession [▶ 60] type exposes the following members.

Events

	Name	Description
	ConnectionStateChanged [▶ 54]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed. (Inherited from IConnectionStateProvider [▶ 52].)

Reference

ISession Interface [▶ 60]

TwinCAT Namespace [▶ 30]

5.1.10 ISessionSettings Interface

Interface ISessionSettings

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISessionSettings
```

VB

```
Public Interface ISessionSettings
```

Reference

TwinCAT Namespace [▶ 30]

5.1.11 ISymbolLoaderSettings Interface

Interface ISymbolLoaderSettings

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISymbolLoaderSettings
```

VB

```
Public Interface ISymbolLoaderSettings
```

Reference

TwinCAT Namespace [▶ 30]

5.1.12 Session Class

Abstract Session base class.

Inheritance Hierarchy

System.Object
  TwinCAT.Session
    TwinCAT.Ads.AdsSession [▶ 347]

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public abstract class Session : ISession,
    IConnectionStateProvider, IDisposable
```

VB

```
Public MustInherit Class Session
    Implements ISession, IConnectionStateProvider, IDisposable
```

The Session type exposes the following members.

Properties

	Name	Description
	AddressSpecifier [▶ 69]	Gets the communication endpoint address string representation.
	Connection [▶ 70]	Gets the (established) connection.
	ConnectionState [▶ 70]	Gets the current Connection state of the Session
	Disposed [▶ 72]	Gets a value indicating whether this Session is disposed.
	EstablishedAt [▶ 72]	Gets the UTC time when the session was established.
	Id [▶ 73]	Gets the Session Identifier
	IsConnected [▶ 73]	Gets a value indicating whether this instance is connected.
	Name [▶ 74]	Gets the name of the session
	SymbolServer [▶ 74]	Gets the symbol server.

Methods

	Name	Description
	Close [▶ 76]	Closes this ISession [▶ 60]
	Connect [▶ 76]	Connects the session.
	Disconnect [▶ 77]	Disconnects the session from the target.
	Dispose. [▶ 78]	Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.
	Dispose(Boolean) [▶ 79]	Releases unmanaged and - optionally - managed resources.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetSessionName [▶ 79]	Gets the name/string identifier of the session.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnConnect [▶ 80]	Handler function connecting the Session.
	OnCreateSymbolServer [▶ 80]	Handler function creating the ISymbolServer [▶ 1649]
	OnDisconnect [▶ 81]	Handler function disconnecting the session.
	OnGetAddress [▶ 81]	Handler function getting the address of the session.
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Events

	Name	Description
	ConnectionStateChanged [▶ 82]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed.

Fields

	Name	Description
	connection [▶ 84]	The (established) connection

Reference

TwinCAT Namespace [▶ 30]

TwinCAT.ISession [▶ 60]

System.IDisposable

5.1.12.1 Session Properties

The Session [▶ 67] type exposes the following members.

Properties

	Name	Description
	AddressSpecifier [▶ 69]	Gets the communication endpoint address string representation.
	Connection [▶ 70]	Gets the (established) connection.
	ConnectionState [▶ 70]	Gets the current Connection state of the Session [▶ 67]
	Disposed [▶ 72]	Gets a value indicating whether this Session [▶ 67] is disposed.
	EstablishedAt [▶ 72]	Gets the UTC time when the session was established.
	Id [▶ 73]	Gets the Session Identifier
	IsConnected [▶ 73]	Gets a value indicating whether this instance is connected.
	Name [▶ 74]	Gets the name of the session
	SymbolServer [▶ 74]	Gets the symbol server.

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.1.1 Session.AddressSpecifier Property

Gets the communication endpoint address string representation.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string AddressSpecifier { get; }
```

VB

```
Public ReadOnly Property AddressSpecifier As String
    Get
```

Property Value

Type: String
The address.

Implements

ISession.AddressSpecifier [▶ 61]

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.1.2 Session.Connection Property

Gets the (established) connection.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IConnection Connection { get; }
```

VB

```
Public ReadOnly Property Connection As IConnection
    Get
```

Property Value

Type: IConnection [▶ 46]
The IConnection [▶ 46] if connection established, or **null** if not connected.

Implements

ISession.Connection [▶ 62]

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.1.3 Session.ConnectionState Property

Gets the current Connection state of the Session [▶ 67]

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ConnectionState ConnectionState { get; }
```

VB

```
Public ReadOnly Property ConnectionState As ConnectionState
    Get
```

Property Value

Type: ConnectionState [▶ 39]
The state of the connection.

Implements

IConnectionStateProvider.ConnectionState [▶ 53]

Remarks

The Connection state changes only if the IConnection [▶ 46] is established / shut down or active communication is triggered by the User of the IConnection [▶ 46] object.

Examples

The following sample shows how to keep the ConnectionState updated by triggering ADS Communication.

Trigger ConnectionState changes in WPF Applications

```
private DispatcherTimer _timer = null;
private AdsSession _session = null;
//private AdsConnection _connection = null;

private void Window_Loaded(object sender, RoutedEventArgs e)
{
    _session = new AdsSession(AmsNetId.Local, 10000);
    IConnection connection = _session.Connect();
    tbConnectionState.Text = connection.ConnectionState.ToString();
    _session.ConnectionStateChanged += _session_ConnectionStateChanged;

    _timer = new DispatcherTimer();
    _timer.Interval = TimeSpan.FromMilliseconds(200);
    _timer.Tick += TimerOnTick;

    _timer.Start();
}

private void Window_Unloaded(object sender, RoutedEventArgs e)
{
    _timer.Stop();
    _session.Dispose();
}

private void _session_ConnectionStateChanged(object sender, TwinCAT.ConnectionStateChangedEventArgs
e)
{
    // ConnectionStateChanged will be triggered by communication Invokes
    tbConnectionState.Text = e.NewState.ToString();
}

private void TimerOnTick(object sender, EventArgs eventArgs)
{
    // The Timer Event will occur here in the UI thread because its an DispatcherTimer event!
    // An active ADS request will trigger Connection State periodically!
    StateInfo stateInfo;
    if (_session.Connection.TryReadState(out stateInfo) == AdsErrorCode.NoError)
    {
    tbAdsState.Text = stateInfo.AdsState.ToString();
    }
    else
    {
    tbAdsState.Text = "Invalid";
    }
}
```

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

Session.ConnectionStateChanged [▶ 82]

5.1.12.1.4 Session.Disposed Property

Gets a value indicating whether this Session [▶ 67] is disposed.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Disposed { get; }
```

VB

```
Public ReadOnly Property Disposed As Boolean
    Get
```

Property Value

Type: Boolean
true if disposed; otherwise, false.

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.1.5 Session.EstablishedAt Property

Gets the UTC time when the session was established.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DateTime EstablishedAt { get; }
```

VB

```
Public ReadOnly Property EstablishedAt As DateTime
    Get
```

Property Value

Type: DateTime
The session established at.

Implements

ISession.EstablishedAt [▶ 62]

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.1.6 Session.Id Property

Gets the Session Identifier

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Id { get; }
```

VB

```
Public ReadOnly Property Id As Integer
    Get
```

Property Value

Type: Int32
The identifier.

Implements

ISession.Id [▶ 63]

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.1.7 Session.IsConnected Property

Gets a value indicating whether this instance is connected.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsConnected { get; }
```

VB

```
Public ReadOnly Property IsConnected As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is connected; otherwise, false.

Implements

ISession.IsConnected [▶ 63]

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.1.8 Session.Name Property

Gets the name of the session

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Name { get; }
```

VB

```
Public ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.1.9 Session.SymbolServer Property

Gets the symbol server.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ISymbolServer SymbolServer { get; }
```

VB

```
Public ReadOnly Property SymbolServer As ISymbolServer
    Get
```

Property Value

Type: ISymbolServer [▶ 1649]
The symbol server.

Remarks

The Session [▶ 67] object holds and caches the symbolic information. To initially create this information, the Connection must be established.

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.2 Session Methods

The Session [▶ 67] type exposes the following members.

Methods

	Name	Description
	Close [▶ 76]	Closes this ISession [▶ 60]
	Connect [▶ 76]	Connects the session.
	Disconnect [▶ 77]	Disconnects the session from the target.
	Dispose. [▶ 78]	Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.
	Dispose(Boolean) [▶ 79]	Releases unmanaged and - optionally - managed resources.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetSessionName [▶ 79]	Gets the name/string identifier of the session.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnConnect [▶ 80]	Handler function connecting the Session.

	Name	Description
	OnCreateSymbolServer [▶ 80]	Handler function creating the ISymbolServer [▶ 1649]
	OnDisconnect [▶ 81]	Handler function disconnecting the session.
	OnGetAddress [▶ 81]	Handler function getting the address of the session.
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.2.1 Session.Close Method

Closes this ISession [▶ 60]

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Close()
```

VB

```
Public Sub Close
```

Implements

ISession.Close. [▶ 64]

Remarks

Closes also the IConnection [▶ 46].

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.2.2 Session.Connect Method

Connects the session.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IConnection Connect()
```

VB

```
Public Function Connect As IConnection
```

Return Value

Type: IConnection [▶ 46]
true if XXXX, false otherwise.

Implements

ISession.Connect. [▶ 65]

Exceptions

Exception	Condition
ObjectDisposedException	
ObjectDisposedException	

Remarks

The IConnection [▶ 46] will be valid until the ISession [▶ 60] is disconnected via the Disconnect. [▶ 77] method or the Dispose method is called. Any possible resurrections after communication losses will be done transparently within the IConnection [▶ 46] so that the IConnection [▶ 46] instance and ISession [▶ 60] instance remains.

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.2.3 Session.Disconnect Method

Disconnects the session from the target.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Disconnect()
```

VB

```
Public Function Disconnect As Boolean
```

Return Value

Type: Boolean
true if XXXX, false otherwise.

Implements

ISession.Disconnect. [▶ 65]

Exceptions

Exception	Condition
ObjectDisposedException	

Remarks

Closes (and disposes) the underlying IConnection [▶ 46] The Session [▶ 67] itself will not be Disposed and can be reconnected.

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.2.4 Session.Dispose Method

Overload List

	Name	Description
	Dispose. [▶ 78]	Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.
	Dispose(Boolean) [▶ 79]	Releases unmanaged and - optionally - managed resources.

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

Session.Dispose Method

Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Dispose()
```

VB

```
Public Sub Dispose
```

Implements

IDisposable.Dispose.

Reference

Session Class [▶ 67]

Dispose Overload [▶ 78]

TwinCAT Namespace [▶ 30]

Session.Dispose Method (Boolean)

Releases unmanaged and - optionally - managed resources.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual void Dispose(
    bool disposing
)
```

VB

```
Protected Overridable Sub Dispose (
    disposing As Boolean
)
```

Parameters

disposing	Type: System.Boolean true to release both managed and unmanaged resources; false to release only unmanaged resources.

Reference

Session Class [▶ 67]

Dispose Overload [▶ 78]

TwinCAT Namespace [▶ 30]

5.1.12.2.5 Session.GetSessionName Method

Gets the name/string identifier of the session.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected abstract string GetSessionName()
```

VB

```
Protected MustOverride Function GetSessionName As String
```

Return Value

Type: String
System.String.

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.2.6 Session.OnConnect Method

Handler function connecting the Session.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual IConnection OnConnect(
    bool reconnect
)
```

VB

```
Protected Overridable Function OnConnect (
    reconnect As Boolean
) As IConnection
```

Parameters

reconnect	Type: System.Boolean if set to true [reconnect].

Return Value

Type: IConnection [▶ 46]
IConnection.

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.2.7 Session.OnCreateSymbolServer Method

Handler function creating the ISymbolServer [▶ 1649]

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected abstract ISymbolServer OnCreateSymbolServer()
```

VB

```
Protected MustOverride Function OnCreateSymbolServer As ISymbolServer
```

Return Value

Type: ISymbolServer [▶ 1649]
ISymbolServer.

Exceptions

Exception	Condition
SessionNotConnectedException [▶ 97]	The connection is not established!

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.2.8 Session.OnDisconnect Method

Handler function disconnecting the session.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual bool OnDisconnect()
```

VB

```
Protected Overridable Function OnDisconnect As Boolean
```

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.2.9 Session.OnGetAddress Method

Handler function getting the address of the session.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected abstract string OnGetAddress()
```

VB

```
Protected MustOverride Function OnGetAddress As String
```

Return Value

Type: String
System.String.

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.3 Session Events

The Session [▶ 67] type exposes the following members.

Events

	Name	Description
	ConnectionStateChanged [▶ 82]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed.

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.3.1 Session.ConnectionStateChanged Event

Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event EventHandler<ConnectionStateChangedEventArgs> ConnectionStateChanged
```

VB

```
Public Event ConnectionStateChanged As EventHandler(Of ConnectionStateChangedEventArgs)
```

Value

Type: System.EventHandler.ConnectionStateChangedEventArgs [▶ 39].

Implements

IConnectionStateProvider.ConnectionStateChanged [▶ 54]

Remarks

The Connection state changes only if the IConnection [▶ 46] is established / shut down or active communication is triggered by the User of the IConnection [▶ 46] object.

Examples

The following sample shows how to keep the ConnectionState [▶ 70] updated by triggering ADS Communication.

Trigger ConnectionState changes in WPF Applications

```
private DispatcherTimer _timer = null;
private AdsSession _session = null;
//private AdsConnection _connection = null;

private void Window_Loaded(object sender, RoutedEventArgs e)
{
    _session = new AdsSession(AmsNetId.Local, 10000);
    IConnection connection = _session.Connect();
    tbConnectionState.Text = connection.ConnectionState.ToString();
    _session.ConnectionStateChanged += _session_ConnectionStateChanged;

    _timer = new DispatcherTimer();
    _timer.Interval = TimeSpan.FromMilliseconds(200);
    _timer.Tick += TimerOnTick;

    _timer.Start();
}

private void Window_Unloaded(object sender, RoutedEventArgs e)
{
    _timer.Stop();
    _session.Dispose();
}

private void _session_ConnectionStateChanged(object sender, TwinCAT.ConnectionStateChangedEventArgs
e)
{
    // ConnectionStateChanged will be triggered by communication Invokes
    tbConnectionState.Text = e.NewState.ToString();
}

private void TimerOnTick(object sender, EventArgs eventArgs)
{
    // The Timer Event will occur here in the UI thread because its an DispatcherTimer event!
    // An active ADS request will trigger Connection State periodically!
    StateInfo stateInfo;
    if (_session.Connection.TryReadState(out stateInfo) == AdsErrorCode.NoError)
    {
    tbAdsState.Text = stateInfo.AdsState.ToString();
    }
    else
    {
    tbAdsState.Text = "Invalid";
    }
}
```

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

Session.ConnectionState [▶ 70]

5.1.12.4 Session Fields

The Session [▶ 67] type exposes the following members.

Fields

	Name	Description
	connection [▶ 84]	The (established) connection

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.12.4.1 Session.connection Field

The (established) connection

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected IConnection connection
```

VB

```
Protected connection As IConnection
```

Field Value

Type: IConnection [▶ 46]

Reference

Session Class [▶ 67]

TwinCAT Namespace [▶ 30]

5.1.13 SessionConnectionStateChangedEventArgs Class

EventArguments for the ConnectionStatusChanged events.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.ConnectionStateChangedEventArgs [▶ 39]
      TwinCAT.SessionConnectionStateChangedEventArgs

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SessionConnectionStateChangedEventArgs : ConnectionStateChangedEventArgs
```

VB

```
Public Class SessionConnectionStateChangedEventArgs
    Inherits ConnectionStateChangedEventArgs
```

The SessionConnectionStateChangedEventArgs type exposes the following members.

Constructors

	Name	Description
	SessionConnectionStateChangedEventArgs(ConnectionStateChangedReason, ConnectionState, ConnectionState, ISession, IConnection) [▶ 87]	Initializes a new instance of the SessionConnectionStateChangedEventArgs class.
	SessionConnectionStateChangedEventArgs(ConnectionStateChangedReason, ConnectionState, ConnectionState, ISession, IConnection, Exception) [▶ 87]	Initializes a new instance of the SessionConnectionStateChangedEventArgs class.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Fields

	Name	Description
	Connection [▶ 89]	The connection

	Name	Description
	Exception [▶ 44]	Exception, (only for Error [▶ 46] (Inherited from ConnectionStateChangedEventArgs [▶ 39].)
	NewState [▶ 44]	New connection state (Inherited from ConnectionStateChangedEventArgs [▶ 39].)
	OldState [▶ 45]	Old connection state (Inherited from ConnectionStateChangedEventArgs [▶ 39].)
	Reason [▶ 45]	Reason for the event (Inherited from ConnectionStateChangedEventArgs [▶ 39].)
	Session [▶ 90]	The session

Reference

TwinCAT Namespace [▶ 30]

TwinCAT.ConnectionStateChangedEventArgs [▶ 39]

5.1.13.1 SessionConnectionStateChangedEventArgs Constructor

Overload List

	Name	Description
	SessionConnectionStateChangedEventArgs(ConnectionStateChangedReason, ConnectionState, ConnectionState, ISession, IConnection) [▶ 87]	Initializes a new instance of the SessionConnectionStateChangedEventArgs [▶ 84] class.
	SessionConnectionStateChangedEventArgs(ConnectionStateChangedReason, ConnectionState, ConnectionState, ISession, IConnection, Exception) [▶ 87]	Initializes a new instance of the SessionConnectionStateChangedEventArgs [▶ 84] class.

Reference

SessionConnectionStateChangedEventArgs Class [▶ 84]

TwinCAT Namespace [▶ 30]

5.1.13.1.1 SessionConnectionStateChangedEventArgs Constructor (ConnectionStateChangedReason, ConnectionState, ConnectionState, ISession, IConnection)

Initializes a new instance of the SessionConnectionStateChangedEventArgs [▶ 84] class.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SessionConnectionStateChangedEventArgs(
    ConnectionStateChangedReason reason,
    ConnectionState newState,
    ConnectionState oldState,
    ISession session,
    IConnection connection
)
```

VB

```
Public Sub New (
    reason As ConnectionStateChangedReason,
    newState As ConnectionState,
    oldState As ConnectionState,
    session As ISession,
    connection As IConnection
)
```

Parameters

reason	Type: TwinCAT.ConnectionStateChangedReason [▶ 46] The reason.
newState	Type: TwinCAT.ConnectionState [▶ 39] The new state.
oldState	Type: TwinCAT.ConnectionState [▶ 39] The old state.
session	Type: TwinCAT.ISession [▶ 60] The session.
connection	Type: TwinCAT.IConnection [▶ 46] The connection.

Reference

SessionConnectionStateChangedEventArgs Class [▶ 84]

SessionConnectionStateChangedEventArgs Overload [▶ 86]

TwinCAT Namespace [▶ 30]

5.1.13.1.2 SessionConnectionStateChangedEventArgs Constructor (ConnectionStateChangedReason, ConnectionState, ConnectionState, ISession, IConnection, Exception)

Initializes a new instance of the SessionConnectionStateChangedEventArgs [▶ 84] class.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SessionConnectionStateChangedEventArgs(
    ConnectionStateChangedReason reason,
    ConnectionState newState,
    ConnectionState oldState,
    ISession session,
    IConnection connection,
    Exception e
)
```

VB

```
Public Sub New (
    reason As ConnectionStateChangedReason,
    newState As ConnectionState,
    oldState As ConnectionState,
    session As ISession,
    connection As IConnection,
    e As Exception
)
```

Parameters

reason	Type: TwinCAT.ConnectionStateChangedReason [▶ 46] The reason.
newState	Type: TwinCAT.ConnectionState [▶ 39] The new state.
oldState	Type: TwinCAT.ConnectionState [▶ 39] The old state.
session	Type: TwinCAT.ISession [▶ 60] The session.
connection	Type: TwinCAT.IConnection [▶ 46] The connection.
e	Type: System.Exception The e.

Reference

SessionConnectionStateChangedEventArgs Class [▶ 84]

SessionConnectionStateChangedEventArgs Overload [▶ 86]

TwinCAT Namespace [▶ 30]

5.1.13.2 SessionConnectionStateChangedEventArgs Methods

The SessionConnectionStateChangedEventArgs [▶ 84] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)

	Name	Description
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

SessionConnectionStateChangedEventArgs Class [▶ 84]

TwinCAT Namespace [▶ 30]

5.1.13.3 SessionConnectionStateChangedEventArgs Fields

The SessionConnectionStateChangedEventArgs [▶ 84] type exposes the following members.

Fields

	Name	Description
	Connection [▶ 89]	The connection
	Exception [▶ 44]	Exception, (only for Error [▶ 46] (Inherited from ConnectionStateChangedEventArgs [▶ 39].)
	NewState [▶ 44]	New connection state (Inherited from ConnectionStateChangedEventArgs [▶ 39].)
	OldState [▶ 45]	Old connection state (Inherited from ConnectionStateChangedEventArgs [▶ 39].)
	Reason [▶ 45]	Reason for the event (Inherited from ConnectionStateChangedEventArgs [▶ 39].)
	Session [▶ 90]	The session

Reference

SessionConnectionStateChangedEventArgs Class [▶ 84]

TwinCAT Namespace [▶ 30]

5.1.13.3.1 SessionConnectionStateChangedEventArgs.Connection Field

The connection

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly IConnection Connection
```

VB

```
Public ReadOnly Connection As IConnection
```

Field Value

Type: IConnection [▶ 46]

Reference

SessionConnectionStateChangedEventArgs Class [▶ 84]

TwinCAT Namespace [▶ 30]

5.1.13.3.2 SessionConnectionStateChangedEventArgs.Session Field

The session

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly ISession Session
```

VB

```
Public ReadOnly Session As ISession
```

Field Value

Type: ISession [▶ 60]

Reference

SessionConnectionStateChangedEventArgs Class [▶ 84]

TwinCAT Namespace [▶ 30]

5.1.14 SessionException Class

Session Exception

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.SessionException
          TwinCAT.SessionNotConnectedException [▶ 97]

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class SessionException : AdsException
```

VB

```
<SerializableAttribute>
Public Class SessionException
    Inherits AdsException
```

The SessionException type exposes the following members.

Constructors

	Name	Description
	SessionException(String, ISession) [▶ 92]	Initializes a new instance of the SessionException class.
	SessionException(String, ISession, Exception) [▶ 93]	Initializes a new instance of the SessionException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData [▶ 95]	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Overrides Exception.GetObjectData(SerializationInfo, StreamingContext).)

	Name	Description
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Fields

	Name	Description
	Session [▶ 96]	The session

Reference

TwinCAT Namespace [▶ 30]

TwinCAT.Ads.AdsException [▶ 318]

5.1.14.1 SessionException Constructor

Overload List

	Name	Description
	SessionException(String, ISession) [▶ 92]	Initializes a new instance of the SessionException [▶ 90] class.
	SessionException(String, ISession, Exception) [▶ 93]	Initializes a new instance of the SessionException [▶ 90] class.

Reference

SessionException Class [▶ 90]

TwinCAT Namespace [▶ 30]

5.1.14.1.1 SessionException Constructor (String, ISession)

Initializes a new instance of the SessionException [▶ 90] class.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SessionException(
    string message,
    ISession session
)
```

VB

```
Public Sub New (
    message As String,
    session As ISession
)
```

Parameters

message	Type: System.String The message.
session	Type: TwinCAT.ISession [▶ 60] The session.

Reference

SessionException Class [▶ 90]

SessionException Overload [▶ 92]

TwinCAT Namespace [▶ 30]

5.1.14.1.2 SessionException Constructor (String, ISession, Exception)

Initializes a new instance of the SessionException [▶ 90] class.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SessionException(
    string message,
    ISession session,
    Exception innerException
)
```

VB

```
Public Sub New (
    message As String,
    session As ISession,
    innerException As Exception
)
```

Parameters

message	Type: System.String The message.
session	Type: TwinCAT.ISession [▶ 60] The session.
innerException	Type: System.Exception The inner exception.

Reference

SessionException Class [▶ 90]

SessionException Overload [▶ 92]

TwinCAT Namespace [▶ 30]

5.1.14.2 SessionException Properties

The SessionException [▶ 90] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

SessionException Class [▶ 90]

TwinCAT Namespace [▶ 30]

5.1.14.3 SessionException Methods

The SessionException [▶ 90] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)

	Name	Description
	GetObjectData [▶ 95]	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Overrides Exception.GetObjectData(SerializationInfo, StreamingContext).)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

SessionException Class [▶ 90]

TwinCAT Namespace [▶ 30]

5.1.14.3.1 SessionException.GetObjectData Method

When overridden in a derived class, sets the SerializationInfo with information about the exception.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override void GetObjectData(
    SerializationInfo info,
    StreamingContext context
)
```

VB

```
Public Overrides Sub GetObjectData (
    info As SerializationInfo,
    context As StreamingContext
)
```

Parameters

info	Type: System.Runtime.Serialization.SerializationInfo The SerializationInfo that holds the serialized object data about the exception being thrown.
context	Type: System.Runtime.Serialization.StreamingContext The StreamingContext that contains contextual information about the source or destination.

Implements

ISerializable.GetObjectData(SerializationInfo, StreamingContext)
_Exception.GetObjectData(SerializationInfo, StreamingContext)

Exceptions

Exception	Condition
ArgumentNullException	info

Reference

SessionException Class [▶ 90]

TwinCAT Namespace [▶ 30]

5.1.14.4 SessionException Events

The SessionException [▶ 90] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

SessionException Class [▶ 90]

TwinCAT Namespace [▶ 30]

5.1.14.5 SessionException Fields

The SessionException [▶ 90] type exposes the following members.

Fields

	Name	Description
	Session [▶ 96]	The session

Reference

SessionException Class [▶ 90]

TwinCAT Namespace [▶ 30]

5.1.14.5.1 SessionException.Session Field

The session

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[NonSerializedAttribute]
public readonly ISession Session
```

VB

```
<NonSerializedAttribute>
Public ReadOnly Session As ISession
```

Field Value

Type: ISession [▶ 60]

Reference

SessionException Class [▶ 90]

TwinCAT Namespace [▶ 30]

5.1.15 SessionNotConnectedException Class

Class SessionNotConnectedException.

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.SessionException [▶ 90]
          TwinCAT.SessionNotConnectedException

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class SessionNotConnectedException : SessionException
```

VB

```
<SerializableAttribute>
Public Class SessionNotConnectedException
    Inherits SessionException
```

The SessionNotConnectedException type exposes the following members.

Constructors

	Name	Description
	SessionNotConnectedException(ISession) [▶ 99]	Initializes a new instance of the SessionNotConnectedException class.
	SessionNotConnectedException(String, ISession) [▶ 99]	Initializes a new instance of the SessionNotConnectedException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)

	Name	Description
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData [▶ 95]	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from SessionException [▶ 90].)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Fields

	Name	Description
	Session [▶ 96]	The session (Inherited from SessionException [▶ 90].)

Reference

TwinCAT Namespace [▶ 30]

TwinCAT.SessionException [▶ 90]

5.1.15.1 SessionNotConnectedException Constructor

Overload List

	Name	Description
	SessionNotConnectedException(ISession) [▶ 99]	Initializes a new instance of the SessionNotConnectedException [▶ 97] class.
	SessionNotConnectedException(String, ISession) [▶ 99]	Initializes a new instance of the SessionNotConnectedException [▶ 97] class.

Reference

SessionNotConnectedException Class [▶ 97]

TwinCAT Namespace [▶ 30]

5.1.15.1.1 SessionNotConnectedException Constructor (ISession)

Initializes a new instance of the SessionNotConnectedException [▶ 97] class.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SessionNotConnectedException(
    ISession session
)
```

VB

```
Public Sub New (
    session As ISession
)
```

Parameters

session	Type: TwinCAT.ISession [▶ 60] The session.

Reference

SessionNotConnectedException Class [▶ 97]

SessionNotConnectedException Overload [▶ 99]

TwinCAT Namespace [▶ 30]

5.1.15.1.2 SessionNotConnectedException Constructor (String, ISession)

Initializes a new instance of the SessionNotConnectedException [▶ 97] class.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SessionNotConnectedException(
    string message,
    ISession session
)
```

VB

```
Public Sub New (
    message As String,
    session As ISession
)
```

Parameters

message	Type: System.String The message.
session	Type: TwinCAT.ISession [▶ 60] The session.

Reference

SessionNotConnectedException Class [▶ 97]

SessionNotConnectedException Overload [▶ 99]

TwinCAT Namespace [▶ 30]

5.1.15.2 SessionNotConnectedException Properties

The SessionNotConnectedException [▶ 97] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

SessionNotConnectedException Class [▶ 97]

TwinCAT Namespace [▶ 30]

5.1.15.3 SessionNotConnectedException Methods

The SessionNotConnectedException [▶ 97] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData [▶ 95]	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from SessionException [▶ 90].)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

SessionNotConnectedException Class [▶ 97]

TwinCAT Namespace [▶ 30]

5.1.15.4 SessionNotConnectedException Events

The SessionNotConnectedException [▶ 97] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

SessionNotConnectedException Class [▶ 97]

TwinCAT Namespace [▶ 30]

5.1.15.5 SessionNotConnectedException Fields

The SessionNotConnectedException [▶ 97] type exposes the following members.

Fields

	Name	Description
	Session [▶ 96]	The session (Inherited from SessionException [▶ 90].)

Reference

SessionNotConnectedException Class [▶ 97]

TwinCAT Namespace [▶ 30]

5.1.16 SessionProvider.S, A, C. Class

Abstract base class for a Custom Session provider

Inheritance Hierarchy

System.Object
  TwinCAT.SessionProvider.S, A, C.
Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public abstract class SessionProvider<S, A, C> : ISessionProvider<S, A, C>
where S : ISession
where C : class
```

VB

```
Public MustInherit Class SessionProvider(Of S As ISession, A, C As Class)
    Implements ISessionProvider(Of S, A, C)
```

Type Parameters

S	SessionType
A	Address type
C	Communication settings type

The SessionProvider.S, A, C. type exposes the following members.

Constructors

	Name	Description
	SessionProvider.S, A, C.. [▶ 104]	Initializes a new instance of the SessionProvider.S, A, C. class.
	SessionProvider.S, A, C. (SessionProviderCapabilities) [▶ 104]	Initializes a new instance of the SessionProvider.S, A, C. class.

Properties

	Name	Description
	Capabilities [▶ 105]	Gets the capabilities.
	Name [▶ 105]	Gets the name of the SessionProvider

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Fields

	Name	Description
	capabilities [▶ 107]	The capabilities of the ISessionProvider

Reference

TwinCAT Namespace [▶ 30]

ISessionProvider

5.1.16.1 SessionProvider.S, A, C. Constructor

Overload List

	Name	Description
	SessionProvider.S, A, C.. [▶ 104]	Initializes a new instance of the SessionProvider.S, A, C. [▶ 102] class.
	SessionProvider.S, A, C. (SessionProviderCapabilities) [▶ 104]	Initializes a new instance of the SessionProvider.S, A, C. [▶ 102] class.

Reference

SessionProvider.S, A, C. Class [▶ 102]

TwinCAT Namespace [▶ 30]

5.1.16.1.1 SessionProvider.S, A, C. Constructor

Initializes a new instance of the SessionProvider.S, A, C. [▶ 102] class.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected SessionProvider()
```

VB

```
Protected Sub New
```

Exceptions

Exception	Condition
Exception	Session provider already instantiated!

Reference

SessionProvider.S, A, C. Class [▶ 102]

SessionProvider.S, A, C. Overload [▶ 103]

TwinCAT Namespace [▶ 30]

5.1.16.1.2 SessionProvider.S, A, C. Constructor (SessionProviderCapabilities)

Initializes a new instance of the SessionProvider.S, A, C. [▶ 102] class.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected SessionProvider(
    SessionProviderCapabilities cap
)
```

VB

```
Protected Sub New (
    cap As SessionProviderCapabilities
)
```

Parameters

cap	Type: TwinCAT.SessionProviderCapabilities [▶ 107]

Exceptions

Exception	Condition
Exception	Session provider already instantiated!

Reference

SessionProvider.S, A, C. Class [▶ 102]

SessionProvider.S, A, C. Overload [▶ 103]

TwinCAT Namespace [▶ 30]

5.1.16.2 SessionProvider.S, A, C. Properties

The SessionProvider.S, A, C. [▶ 102] generic type exposes the following members.

Properties

	Name	Description
	Capabilities [▶ 105]	Gets the capabilities.
	Name [▶ 105]	Gets the name of the SessionProvider

Reference

SessionProvider.S, A, C. Class [▶ 102]

TwinCAT Namespace [▶ 30]

5.1.16.2.1 SessionProvider.S, A, C..Capabilities Property

Gets the capabilities.

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SessionProviderCapabilities Capabilities { get; }
```

VB

```
Public ReadOnly Property Capabilities As SessionProviderCapabilities
    Get
```

Property Value

Type: SessionProviderCapabilities [▶ 107]
The capabilities.

Reference

SessionProvider.S, A, C. Class [▶ 102]

TwinCAT Namespace [▶ 30]

5.1.16.2.2 SessionProvider.S, A, C..Name Property

Gets the name of the SessionProvider

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public abstract string Name { get; }
```

VB

```
Public MustOverride ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Reference

SessionProvider.S, A, C. Class [▶ 102]

TwinCAT Namespace [▶ 30]

5.1.16.3 SessionProvider.S, A, C. Methods

The SessionProvider.S, A, C. [▶ 102] generic type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

SessionProvider.S, A, C. Class [▶ 102]

TwinCAT Namespace [▶ 30]

5.1.16.4 SessionProvider.S, A, C. Fields

The SessionProvider.S, A, C. [▶ 102] generic type exposes the following members.

Fields

	Name	Description
	capabilities [▶ 107]	The capabilities of the ISessionProvider

Reference

SessionProvider.S, A, C. Class [▶ 102]

TwinCAT Namespace [▶ 30]

5.1.16.4.1 SessionProvider.S, A, C..capabilities Field

The capabilities of the ISessionProvider

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected SessionProviderCapabilities capabilities
```

VB

```
Protected capabilities As SessionProviderCapabilities
```

Field Value

Type: SessionProviderCapabilities [▶ 107]

Reference

SessionProvider.S, A, C. Class [▶ 102]

TwinCAT Namespace [▶ 30]

5.1.17 SessionProviderCapabilities Enumeration

Enum SessionProviderCapabilities

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[FlagsAttribute]
public enum SessionProviderCapabilities
```

VB

```
<FlagsAttribute>
Public Enumeration SessionProviderCapabilities
```

Members

	Member name	Value	Description
	DataTypeSupport	1	Supports DataTypes
	SymbolBrowsing	2	Supports Symbol Browsing
	ValueRead	4	Supports VAlue Read
	ValueWrite	8	Support Value Write
	ValueNotifications	16	Supports Value changed Notifications
	None	0	Uninitilialized / None
	Mask_All	31	All Capabilities active

Reference

TwinCAT Namespace [▶ 30]

5.1.18 SymbolsLoadMode Enumeration

Enum SymbolsLoadMode

Namespace: TwinCAT [▶ 30]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public enum SymbolsLoadMode
```

VB

```
Public Enumeration SymbolsLoadMode
```

Members

	Member name	Value	Description
	Flat	0	Loads the Symbols organized as Flat List
	VirtualTree	1	Loads the Symbols organized as Virtual tree (Symbol Parent - Child relationships)
	DynamicTree	2	Loads the Symbols as a Virtual tree with Dynamic Symbols (Only available within versions > 4.X of this ADS Api)

Reference

TwinCAT Namespace [▶ 30]

5.2 TwinCAT.Ads Namespace

ADS root namespace.

Classes

	Class	Description
	AdsBinaryReader [▶ 112]	Derives from BinaryReader and reads primitive as well as PLC data types as binary values.
	AdsBinaryWriter [▶ 121]	Derives from BinaryWriter and writes primitive and PLC data types in binary to a stream.
	AdsClientSettings [▶ 133]	Settings object for TcAdsClient [▶ 625].
	AdsCommunicationStatistics [▶ 139]	ADS Communication statistics
	AdsConnection [▶ 148]	ADS Connection class
	AdsDatatypeArrayInfo [▶ 296]	Array definition for a single dimension.
	AdsDatatypeNotSupportedException [▶ 300]	The exception that is thrown when a ADS datatype is not supported.
	AdsErrorException [▶ 310]	The exception that is thrown when an ADS error occurs.
	AdsException [▶ 318]	Base class for all exceptions thrown by this class.
	AdsInitializeException [▶ 324]	Initializing exception (TcAdsDllCe resp. TcAdsDll.dll not found)
	AdsInvalidNotificationException [▶ 328]	This AdsInvalidNotificationException is created if the length of the notification data is 0. This indicates that the notification handle is not valid any more. This exception is passed to the AdsNotificationErrorEvent.
	AdsNotificationErrorEventArgs [▶ 333]	Provides data for AdsNotificationErrorEvent of the class TcAdsClient.
	AdsNotificationEventArgs [▶ 336]	Provides data for AdsNotificationEvent of the class TcAdsClient.
	AdsNotificationExEventArgs [▶ 342]	Provides data for AdsNotificationExEvent of the class TcAdsClient.
	AdsSession [▶ 347]	AdsSession class
	AdsStateChangedEventArgs [▶ 366]	Provides data for AdsStateChangedEvent of the class TcAdsClient.
	AdsStateChangedEventArgs2 [▶ 370]	Event Arguments for AdsStateChanged events.
	AdsStream [▶ 375]	The class AdsStream is a stream class used for ADS communication.

	Class	Description
	AdsSumCommandException [▶ 384]	The exception that is thrown when an ADS SumCommand error occurs.
	AdsSymbolException [▶ 389]	Symbol Exception
	AdsSymbolVersionChangedEventArgs [▶ 398]	Provides data for AdsSymbolVersionChangedEvent of the class TcAdsClient.
	AmsAddress [▶ 410]	Ams/Ads Address
	AmsNetId [▶ 428]	AMS/ADS Net ID
	AmsRouterNotificationEventArgs [▶ 452]	Provides data for AmsRouterNotificationEvent of the class TcAdsClient.
	NotificationSettings [▶ 573]	Notification communication settings
	ReadOnlyTcAdsDataTypeCollection [▶ 581]	Read only collection of ITcAdsDataType [▶ 524]" types.
	RpcMethodNotSupportedException [▶ 586]	Symbol Exception
	SessionSettings [▶ 592]	Session settings class
	SymbolException [▶ 606]	Symbol bound exceptions
	SymbolLoaderSettings [▶ 615]	Settings object for the IAdsSymbolLoader [▶ 979] initialization.
	TcAdsClient [▶ 625]	ADS Client / ADS Communication object.
	TcAdsSymbolInfo [▶ 780]	The class TcAdsSymbolInfo represents a symbol loaded by an instance of the TcAdsSymbolInfoLoader class.
	TcAdsSymbolInfoCollection [▶ 805]	Represents a collection of TcAdsSymbolInfo objects.
	TcAdsSymbolInfoLoader [▶ 812]	The class TcAdsSymbolInfoLoader is responsible for downloading the list of declared variables and the data types from an ADS Server.

Structures

	Structure	Description
	AdsVersion [▶ 405]	The structure contains the version number, revision number and build number.

	Structure	Description
	DeviceInfo [▶ 455]	The structure contains the name and the version information of the device.
	StateInfo [▶ 598]	The structure contains the ADS state and device state.

Interfaces

	Interface	Description
	IAdsAnyAccess [▶ 458]	Interface for accessing ADS 'Any' objects.
	IAdsConnection [▶ 469]	ADS Connection interface
	IAdsHandleAccess [▶ 485]	Interface for ads access via variable handle
	IAdsNotifications [▶ 496]	Interface for Notification management.
	IAdsSession [▶ 516]	Interface IAdsSession
	IAdsSessionSettings [▶ 521]	Interface for ADS Session Settings
	IFailFastHandler [▶ 523]	Interface for a fast failing (Circuit breaker) ads handler
	ITcAdsDataType [▶ 524]	Interface ITcAdsDataType
	ITcAdsRpcInvoke [▶ 534]	Interface ITcAdsRpcInvoke
	ITcAdsSubItem [▶ 544]	Interface ITcAdsSubItem
	ITcAdsSymbol [▶ 549]	Defines an Interface for reading the ADS symbol information.
	ITcAdsSymbol2 [▶ 553]	Interface ITcAdsSymbol2 (extends ITcAdsSymbol [▶ 549])
	ITcAdsSymbol3 [▶ 558]	Interface ITcAdsSymbol3 (extends ITcAdsSymbol [▶ 549] ... ITcAdsSymbol3 [▶ 558])
	ITcAdsSymbol4 [▶ 561]	Interface ITcAdsSymbol4 (extends ITcAdsSymbol [▶ 549] ... ITcAdsSymbol3 [▶ 558]
	ITcAdsSymbol5 [▶ 566]	Interface ITcAdsSymbol5 (extends ITcAdsSymbol [▶ 549] ... ITcAdsSymbol4 [▶ 561]
	ITcAdsSymbolBrowser [▶ 572]	Interface ITcAdsSubSymbolProvider

Delegates

	Delegate	Description
	AdsNotificationErrorEventHandler [▶ 335]	Event handler for the AdsNotificationError event in the class TcAdsClient.

	Delegate	Description
	AdsNotificationEventHandler [▶ 342]	Event handler for the AdsNotification event in the class TcAdsClient.
	AdsNotificationExEventHandler [▶ 346]	Event handler for the AdsNotification event in the class TcAdsClient.
	AdsStateChangedEventHandler [▶ 374]	Event handler for the AdsStateChanged event in the class TcAdsClient.
	AmsRouterNotificationEventHandler [▶ 454]	Event handler for the AmsRouterNotification event in the class TcAdsClient.

Enumerations

	Enumeration	Description
	AdsDatatypeId [▶ 299]	ADS data types.
	AdsErrorCode [▶ 305]	Describes the ADS error that occured.
	AdsState [▶ 365]	Describes the AdsState.
	AdsTransMode [▶ 401]	ADS Transmission Mode for ADS Notifications.
	AmsPort [▶ 450]	AmsPorts
	AmsRouterState [▶ 455]	State of the AMS Router.
	TransportProtocol [▶ 820]	Enum ADS TransportProtocol

5.2.1 AdsBinaryReader Class

Derives from BinaryReader and reads primitive as well as PLC data types as binary values.

Inheritance Hierarchy

System.Object
  System.IO.BinaryReader
    TwinCAT.Ads.AdsBinaryReader

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class AdsBinaryReader : BinaryReader
```

VB

```
Public Class AdsBinaryReader
    Inherits BinaryReader
```

The AdsBinaryReader type exposes the following members.

Constructors

	Name	Description
	AdsBinaryReader [▶ 115]	Initializes a new instance of the AdsBinaryReader class based on the supplied stream.

Properties

	Name	Description
	BaseStream	Exposes access to the underlying stream of the BinaryReader. (Inherited from BinaryReader.)

Methods

	Name	Description
	Close	Closes the current reader and the underlying stream. (Inherited from BinaryReader.)
	Dispose.	Releases all resources used by the current instance of the BinaryReader class. (Inherited from BinaryReader.)
	Dispose(Boolean)	Releases the unmanaged resources used by the BinaryReader class and optionally releases the managed resources. (Inherited from BinaryReader.)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	FillBuffer	Fills the internal buffer with the specified number of bytes read from the stream. (Inherited from BinaryReader.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	PeekChar	Returns the next available character and does not advance the byte or character position. (Inherited from BinaryReader.)
	Read.	Reads characters from the underlying stream and advances the current position of the stream in accordance with the Encoding used and the specific character being read from the stream. (Inherited from BinaryReader.)
	Read(.Char., Int32, Int32)	Reads the specified number of characters from the stream, starting from a specified point in the character array. (Inherited from BinaryReader.)
	Read(.Byte., Int32, Int32)	Reads the specified number of bytes from the stream, starting from a specified point in the byte array. (Inherited from BinaryReader.)
	Read7BitEncodedInt	Reads in a 32-bit integer in compressed format. (Inherited from BinaryReader.)
	ReadBoolean	Reads a Boolean value from the current stream and advances the current position of the stream by one byte. (Inherited from BinaryReader.)
	ReadByte	Reads the next byte from the current stream and advances the current position of the stream by one byte. (Inherited from BinaryReader.)

	Name	Description
	ReadBytes	Reads the specified number of bytes from the current stream into a byte array and advances the current position by that number of bytes. (Inherited from BinaryReader.)
	ReadChar	Reads the next character from the current stream and advances the current position of the stream in accordance with the Encoding used and the specific character being read from the stream. (Inherited from BinaryReader.)
	ReadChars	Reads the specified number of characters from the current stream, returns the data in a character array, and advances the current position in accordance with the Encoding used and the specific character being read from the stream. (Inherited from BinaryReader.)
	ReadDecimal	Reads a decimal value from the current stream and advances the current position of the stream by sixteen bytes. (Inherited from BinaryReader.)
	ReadDouble	Reads an 8-byte floating point value from the current stream and advances the current position of the stream by eight bytes. (Inherited from BinaryReader.)
	ReadGuid [▶ 117]	Reads a Guid from the current stream.
	ReadInt16	Reads a 2-byte signed integer from the current stream and advances the current position of the stream by two bytes. (Inherited from BinaryReader.)
	ReadInt32	Reads a 4-byte signed integer from the current stream and advances the current position of the stream by four bytes. (Inherited from BinaryReader.)
	ReadInt64	Reads an 8-byte signed integer from the current stream and advances the current position of the stream by eight bytes. (Inherited from BinaryReader.)
	ReadPlcAnsiString [▶ 118]	Reads a PLC string from the current stream (Encoding.Default, ANSI Encoding)
	ReadPlcDATE [▶ 119]	Reads a PLC Date type from the current stream.
	ReadPlcTIME [▶ 120]	Reads a PLC 'TIME' data type from the current stream.
	ReadPlcUnicodeString [▶ 120]	Reads a PLC string from the current stream (Encoding.Unicode, Unicode Encoding)
	ReadSByte	Reads a signed byte from this stream and advances the current position of the stream by one byte. (Inherited from BinaryReader.)
	ReadSingle	Reads a 4-byte floating point value from the current stream and advances the current position of the stream by four bytes. (Inherited from BinaryReader.)
	ReadString	Reads a string from the current stream. The string is prefixed with the length, encoded as an integer seven bits at a time. (Inherited from BinaryReader.)
	ReadUInt16	Reads a 2-byte unsigned integer from the current stream using little-endian encoding and advances the position of the stream by two bytes. (Inherited from BinaryReader.)
	ReadUInt32	Reads a 4-byte unsigned integer from the current stream and advances the position of the stream by four bytes. (Inherited from BinaryReader.)
	ReadUInt64	Reads an 8-byte unsigned integer from the current stream and advances the position of the stream by eight bytes. (Inherited from BinaryReader.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Remarks

The AdsBinaryReader object is initialized with System.Text.Encoding.Default (ANSI Encoding).

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.1.1 AdsBinaryReader Constructor

Initializes a new instance of the AdsBinaryReader class based on the supplied stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsBinaryReader(
    AdsStream stream
)
```

VB

```
Public Sub New (
    stream As AdsStream
)
```

Parameters

stream	Type: TwinCAT.Ads.AdsStream [▶ 375] A stream.

Reference

AdsBinaryReader Class [▶ 112]

TwinCAT.Ads Namespace [▶ 108]

5.2.1.2 AdsBinaryReader Properties

The AdsBinaryReader [▶ 112] type exposes the following members.

Properties

	Name	Description
	BaseStream	Exposes access to the underlying stream of the BinaryReader. (Inherited from BinaryReader.)

Reference

AdsBinaryReader Class [▶ 112]

TwinCAT.Ads Namespace [▶ 108]

5.2.1.3 AdsBinaryReader Methods

The AdsBinaryReader [▶ 112] type exposes the following members.

Methods

	Name	Description
	Close	Closes the current reader and the underlying stream. (Inherited from BinaryReader.)
	Dispose.	Releases all resources used by the current instance of the BinaryReader class. (Inherited from BinaryReader.)
	Dispose(Boolean)	Releases the unmanaged resources used by the BinaryReader class and optionally releases the managed resources. (Inherited from BinaryReader.)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	FillBuffer	Fills the internal buffer with the specified number of bytes read from the stream. (Inherited from BinaryReader.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	PeekChar	Returns the next available character and does not advance the byte or character position. (Inherited from BinaryReader.)
	Read.	Reads characters from the underlying stream and advances the current position of the stream in accordance with the Encoding used and the specific character being read from the stream. (Inherited from BinaryReader.)
	Read(.Char., Int32, Int32)	Reads the specified number of characters from the stream, starting from a specified point in the character array. (Inherited from BinaryReader.)
	Read(.Byte., Int32, Int32)	Reads the specified number of bytes from the stream, starting from a specified point in the byte array. (Inherited from BinaryReader.)
	Read7BitEncodedInt	Reads in a 32-bit integer in compressed format. (Inherited from BinaryReader.)
	ReadBoolean	Reads a Boolean value from the current stream and advances the current position of the stream by one byte. (Inherited from BinaryReader.)
	ReadByte	Reads the next byte from the current stream and advances the current position of the stream by one byte. (Inherited from BinaryReader.)
	ReadBytes	Reads the specified number of bytes from the current stream into a byte array and advances the current position by that number of bytes. (Inherited from BinaryReader.)
	ReadChar	Reads the next character from the current stream and advances the current position of the stream in accordance with the Encoding used and the specific character being read from the stream. (Inherited from BinaryReader.)
	ReadChars	Reads the specified number of characters from the current stream, returns the data in a character array, and advances the current position in accordance with the Encoding used and the specific character being read from the stream. (Inherited from BinaryReader.)
	ReadDecimal	Reads a decimal value from the current stream and advances the current position of the stream by sixteen bytes. (Inherited from BinaryReader.)

	Name	Description
	ReadDouble	Reads an 8-byte floating point value from the current stream and advances the current position of the stream by eight bytes. (Inherited from BinaryReader.)
	ReadGuid [▶ 117]	Reads a Guid from the current stream.
	ReadInt16	Reads a 2-byte signed integer from the current stream and advances the current position of the stream by two bytes. (Inherited from BinaryReader.)
	ReadInt32	Reads a 4-byte signed integer from the current stream and advances the current position of the stream by four bytes. (Inherited from BinaryReader.)
	ReadInt64	Reads an 8-byte signed integer from the current stream and advances the current position of the stream by eight bytes. (Inherited from BinaryReader.)
	ReadPlcAnsiString [▶ 118]	Reads a PLC string from the current stream (Encoding.Default, ANSI Encoding)
	ReadPlcDATE [▶ 119]	Reads a PLC Date type from the current stream.
	ReadPlcTIME [▶ 120]	Reads a PLC 'TIME' data type from the current stream.
	ReadPlcUnicodeString [▶ 120]	Reads a PLC string from the current stream (Encoding.Unicode, Unicode Encoding)
	ReadSByte	Reads a signed byte from this stream and advances the current position of the stream by one byte. (Inherited from BinaryReader.)
	ReadSingle	Reads a 4-byte floating point value from the current stream and advances the current position of the stream by four bytes. (Inherited from BinaryReader.)
	ReadString	Reads a string from the current stream. The string is prefixed with the length, encoded as an integer seven bits at a time. (Inherited from BinaryReader.)
	ReadUInt16	Reads a 2-byte unsigned integer from the current stream using little-endian encoding and advances the position of the stream by two bytes. (Inherited from BinaryReader.)
	ReadUInt32	Reads a 4-byte unsigned integer from the current stream and advances the position of the stream by four bytes. (Inherited from BinaryReader.)
	ReadUInt64	Reads an 8-byte unsigned integer from the current stream and advances the position of the stream by eight bytes. (Inherited from BinaryReader.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

AdsBinaryReader Class [▶ 112]

TwinCAT.Ads Namespace [▶ 108]

5.2.1.3.1 AdsBinaryReader.ReadGuid Method

Reads a Guid from the current stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Guid ReadGuid()
```

VB

```
Public Function ReadGuid As Guid
```

Return Value

Type: Guid
Guid.

Reference

AdsBinaryReader Class [▶ 112]

TwinCAT.Ads Namespace [▶ 108]

5.2.1.3.2 AdsBinaryReader.ReadPlcAnsiString Method

Reads a PLC string from the current stream (Encoding.Default, ANSI Encoding)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ReadPlcAnsiString(
    int byteLength
)
```

VB

```
Public Function ReadPlcAnsiString (
    byteLength As Integer
) As String
```

Parameters

byteLength	Type: System.Int32 The length of the string in the PLC (byte length equals character count on PLC + '\0')

Return Value

Type: String
The string being read (until the first '\0' character)

Remarks

The byte length of a STRING[80] in the PLC is 81. The byte length of a WSTRING[80] in the PLC is 162. Because of ANSI Encoding the number of Chars could differ with the number of Bytes (e.g on Double Byte Codepages DBCS, Codepage 932, Japan)

Examples

The following code shows how to Read/Write ANSI string values..

Read/Write Strings

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local port 851 (PLC)

    int handle = client.CreateVariableHandle("MAIN.string"); // Symbol "string" in MAIN defined as s
tring

    try
    {
    // Read ANSI String string[80]
    int byteSize = 81; // Size of 80 ANSI chars + /0 (STRING[80])
    AdsStream readStream = new AdsStream(byteSize); // Size of 80 ANSI chars + /0 (STRING[80])
    AdsBinaryReader reader = new AdsBinaryReader(readStream);
    client.Read(handle, readStream, 0, byteSize); // Read 81 bytes
    string value = reader.ReadPlcString(byteSize,Encoding.Default);

    // Write ANSI String string[80]
    AdsStream writeStream = new AdsStream(byteSize);
    AdsBinaryWriter writer = new AdsBinaryWriter(writeStream);
    value = "Changed";
    writer.WritePlcString(value, 80,Encoding.Default); // Max 80 characters!
    client.Write(handle, writeStream, 0, byteSize);
    }
    finally
    {
    client.DeleteVariableHandle(handle);
    }
}
```

Reference

AdsBinaryReader Class [▶ 112]

TwinCAT.Ads Namespace [▶ 108]

5.2.1.3.3 AdsBinaryReader.ReadPlcDATE Method

Reads a PLC Date type from the current stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DateTime ReadPlcDATE()
```

VB

```
Public Function ReadPlcDATE As DateTime
```

Return Value

Type: DateTime
The date being read.

Reference

AdsBinaryReader Class [▶ 112]

TwinCAT.Ads Namespace [▶ 108]

5.2.1.3.4 AdsBinaryReader.ReadPlcTIME Method

Reads a PLC 'TIME' data type from the current stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TimeSpan ReadPlcTIME()
```

VB

```
Public Function ReadPlcTIME As TimeSpan
```

Return Value

Type: TimeSpan
The time being read as TimeSpan.

Remarks

This method reads the 4 Byte Plc 'TIME' datatype.

Reference

AdsBinaryReader Class [▶ 112]

TwinCAT.Ads Namespace [▶ 108]

5.2.1.3.5 AdsBinaryReader.ReadPlcUnicodeString Method

Reads a PLC string from the current stream (Encoding.Unicode, Unicode Encoding)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ReadPlcUnicodeString(
    int byteLength
)
```

VB

```
Public Function ReadPlcUnicodeString (
    byteLength As Integer
) As String
```

Parameters

byteLength	Type: System.Int32 The length of the string in the PLC (byte length equals character count on PLC + '\0')

Return Value

Type: String
The string being read (until the first '\0' character)

Remarks

The byte length of a STRING[80] in the PLC is 81. The byte length of a WSTRING[80] in the PLC is 162. Because of ANSI Encoding the number of Chars could differ with the number of Bytes (e.g on Double Byte Codepages DBCS, Codpage 932, Japan)

Examples

The following code shows how to Read/Write UNICODE string values..

Read/Write Unicode Strings

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local port 851 (PLC)

    int handle = client.CreateVariableHandle("MAIN.wstring"); // Symbol "wstring" defined in MAIN as
 WSTRING

    try
    {
    // Read UNICODE String wstring[80]
    int byteSize = 2 * 81; // Size of 80 UNICODE chars + /0 (WSTRING[80])
    AdsStream readStream = new AdsStream(byteSize);
    AdsBinaryReader reader = new AdsBinaryReader(readStream);
    client.Read(handle, readStream, 0, byteSize); // Read 2*81 bytes
    string value = reader.ReadPlcString(byteSize,Encoding.Unicode);

    // Write ANSI String string[80]
    AdsStream writeStream = new AdsStream(byteSize);
    AdsBinaryWriter writer = new AdsBinaryWriter(writeStream);
    value = "Changed";
    writer.WritePlcString(value, 80,Encoding.Unicode);
    client.Write(handle, writeStream, 0, byteSize);
    }
    finally
    {
    client.DeleteVariableHandle(handle);
    }
}
```

Reference

AdsBinaryReader Class [▶ 112]

TwinCAT.Ads Namespace [▶ 108]

5.2.2 AdsBinaryWriter Class

Derives from BinaryWriter and writes primitive and PLC data types in binary to a stream.

Inheritance Hierarchy

System.Object
  System.IO.BinaryWriter
    TwinCAT.Ads.AdsBinaryWriter

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class AdsBinaryWriter : BinaryWriter
```

VB

```
Public Class AdsBinaryWriter
    Inherits BinaryWriter
```

The AdsBinaryWriter type exposes the following members.

Constructors

	Name	Description
	AdsBinaryWriter [▶ 124]	Initializes a new instance of the AdsBinaryWriter class based on the supplied stream.

Properties

	Name	Description
	BaseStream	Gets the underlying stream of the BinaryWriter. (Inherited from BinaryWriter.)

Methods

	Name	Description
	Close	Closes the current BinaryWriter and the underlying stream. (Inherited from BinaryWriter.)
	Dispose.	Releases all resources used by the current instance of the BinaryWriter class. (Inherited from BinaryWriter.)
	Dispose(Boolean)	Releases the unmanaged resources used by the BinaryWriter and optionally releases the managed resources. (Inherited from BinaryWriter.)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	Flush	Clears all buffers for the current writer and causes any buffered data to be written to the underlying device. (Inherited from BinaryWriter.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Seek	Sets the position within the current stream. (Inherited from BinaryWriter.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	Write(Boolean)	Writes a one-byte Boolean value to the current stream, with 0 representing false and 1 representing true. (Inherited from BinaryWriter.)
	Write(Byte)	Writes an unsigned byte to the current stream and advances the stream position by one byte. (Inherited from BinaryWriter.)
	Write(SByte)	Writes a signed byte to the current stream and advances the stream position by one byte. (Inherited from BinaryWriter.)

	Name	Description
	Write(.Byte.)	Writes a byte array to the underlying stream. (Inherited from BinaryWriter.)
	Write(Char)	Writes a Unicode character to the current stream and advances the current position of the stream in accordance with the Encoding used and the specific characters being written to the stream. (Inherited from BinaryWriter.)
	Write(.Char.)	Writes a character array to the current stream and advances the current position of the stream in accordance with the Encoding used and the specific characters being written to the stream. (Inherited from BinaryWriter.)
	Write(Double)	Writes an eight-byte floating-point value to the current stream and advances the stream position by eight bytes. (Inherited from BinaryWriter.)
	Write(Decimal)	Writes a decimal value to the current stream and advances the stream position by sixteen bytes. (Inherited from BinaryWriter.)
	Write(Int16)	Writes a two-byte signed integer to the current stream and advances the stream position by two bytes. (Inherited from BinaryWriter.)
	Write(UInt16)	Writes a two-byte unsigned integer to the current stream and advances the stream position by two bytes. (Inherited from BinaryWriter.)
	Write(Int32)	Writes a four-byte signed integer to the current stream and advances the stream position by four bytes. (Inherited from BinaryWriter.)
	Write(UInt32)	Writes a four-byte unsigned integer to the current stream and advances the stream position by four bytes. (Inherited from BinaryWriter.)
	Write(Int64)	Writes an eight-byte signed integer to the current stream and advances the stream position by eight bytes. (Inherited from BinaryWriter.)
	Write(UInt64)	Writes an eight-byte unsigned integer to the current stream and advances the stream position by eight bytes. (Inherited from BinaryWriter.)
	Write(Single)	Writes a four-byte floating-point value to the current stream and advances the stream position by four bytes. (Inherited from BinaryWriter.)
	Write(String)	Writes a length-prefixed string to this stream in the current encoding of the BinaryWriter, and advances the current position of the stream in accordance with the encoding used and the specific characters being written to the stream. (Inherited from BinaryWriter.)
	Write(.Byte., Int32, Int32)	Writes a region of a byte array to the current stream. (Inherited from BinaryWriter.)
	Write(.Char., Int32, Int32)	Writes a section of a character array to the current stream, and advances the current position of the stream in accordance with the Encoding used and perhaps the specific characters being written to the stream. (Inherited from BinaryWriter.)
	Write7BitEncodedInt	Writes a 32-bit integer in a compressed format. (Inherited from BinaryWriter.)
	WriteGuid [▶ 127]	Writes the Guid (16 Bytes) to the current stream.
	WritePlcAnsiString [▶ 127]	Writes a string as a PLC string to the current stream.
	WritePlcAnsiStringFixedLength [▶ 129]	Writes the PLC ANSI string in a data block of the specified size.
	WritePlcType(DateTime) [▶ 130]	Writes a date as PLC date type to the current stream.

	Name	Description
	WritePlcType(TimeSpan) [▶ 130]	Writes a time span as PLC time type to the current stream.
	WritePlcUnicodeString [▶ 131]	Writes a (unicode) string as a PLC string to the current stream.
	WritePlcUnicodeStringFixedLength [▶ 132]	Writes the PLC UNICODE string into a data block of the specified size.

Fields

	Name	Description
	OutStream	Holds the underlying stream. (Inherited from BinaryWriter.)

Remarks

The AdsBinaryWriter is fixed to use Default encoding (ANSI).

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.2.1 AdsBinaryWriter Constructor

Initializes a new instance of the AdsBinaryWriter class based on the supplied stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsBinaryWriter(
    AdsStream stream
)
```

VB

```
Public Sub New (
    stream As AdsStream
)
```

Parameters

stream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream

Reference

AdsBinaryWriter Class [▶ 121]

TwinCAT.Ads Namespace [▶ 108]

5.2.2.2 AdsBinaryWriter Properties

The AdsBinaryWriter [▶ 121] type exposes the following members.

Properties

	Name	Description
	BaseStream	Gets the underlying stream of the BinaryWriter. (Inherited from BinaryWriter.)

Reference

AdsBinaryWriter Class [▶ 121]

TwinCAT.Ads Namespace [▶ 108]

5.2.2.3 AdsBinaryWriter Methods

The AdsBinaryWriter [▶ 121] type exposes the following members.

Methods

	Name	Description
	Close	Closes the current BinaryWriter and the underlying stream. (Inherited from BinaryWriter.)
	Dispose.	Releases all resources used by the current instance of the BinaryWriter class. (Inherited from BinaryWriter.)
	Dispose(Boolean)	Releases the unmanaged resources used by the BinaryWriter and optionally releases the managed resources. (Inherited from BinaryWriter.)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	Flush	Clears all buffers for the current writer and causes any buffered data to be written to the underlying device. (Inherited from BinaryWriter.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Seek	Sets the position within the current stream. (Inherited from BinaryWriter.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	Write(Boolean)	Writes a one-byte Boolean value to the current stream, with 0 representing false and 1 representing true. (Inherited from BinaryWriter.)
	Write(Byte)	Writes an unsigned byte to the current stream and advances the stream position by one byte. (Inherited from BinaryWriter.)
	Write(SByte)	Writes a signed byte to the current stream and advances the stream position by one byte. (Inherited from BinaryWriter.)
	Write(.Byte.)	Writes a byte array to the underlying stream. (Inherited from BinaryWriter.)

	Name	Description
	Write(Char)	Writes a Unicode character to the current stream and advances the current position of the stream in accordance with the Encoding used and the specific characters being written to the stream. (Inherited from BinaryWriter.)
	Write(.Char.)	Writes a character array to the current stream and advances the current position of the stream in accordance with the Encoding used and the specific characters being written to the stream. (Inherited from BinaryWriter.)
	Write(Double)	Writes an eight-byte floating-point value to the current stream and advances the stream position by eight bytes. (Inherited from BinaryWriter.)
	Write(Decimal)	Writes a decimal value to the current stream and advances the stream position by sixteen bytes. (Inherited from BinaryWriter.)
	Write(Int16)	Writes a two-byte signed integer to the current stream and advances the stream position by two bytes. (Inherited from BinaryWriter.)
	Write(UInt16)	Writes a two-byte unsigned integer to the current stream and advances the stream position by two bytes. (Inherited from BinaryWriter.)
	Write(Int32)	Writes a four-byte signed integer to the current stream and advances the stream position by four bytes. (Inherited from BinaryWriter.)
	Write(UInt32)	Writes a four-byte unsigned integer to the current stream and advances the stream position by four bytes. (Inherited from BinaryWriter.)
	Write(Int64)	Writes an eight-byte signed integer to the current stream and advances the stream position by eight bytes. (Inherited from BinaryWriter.)
	Write(UInt64)	Writes an eight-byte unsigned integer to the current stream and advances the stream position by eight bytes. (Inherited from BinaryWriter.)
	Write(Single)	Writes a four-byte floating-point value to the current stream and advances the stream position by four bytes. (Inherited from BinaryWriter.)
	Write(String)	Writes a length-prefixed string to this stream in the current encoding of the BinaryWriter, and advances the current position of the stream in accordance with the encoding used and the specific characters being written to the stream. (Inherited from BinaryWriter.)
	Write(.Byte., Int32, Int32)	Writes a region of a byte array to the current stream. (Inherited from BinaryWriter.)
	Write(.Char., Int32, Int32)	Writes a section of a character array to the current stream, and advances the current position of the stream in accordance with the Encoding used and perhaps the specific characters being written to the stream. (Inherited from BinaryWriter.)
	Write7BitEncodedInt	Writes a 32-bit integer in a compressed format. (Inherited from BinaryWriter.)
	WriteGuid [▶ 127]	Writes the Guid (16 Bytes) to the current stream.
	WritePlcAnsiString [▶ 127]	Writes a string as a PLC string to the current stream.
	WritePlcAnsiStringFixedLength [▶ 129]	Writes the PLC ANSI string in a data block of the specified size.
	WritePlcType(DateTime) [▶ 130]	Writes a date as PLC date type to the current stream.

	Name	Description
	WritePlcType(TimeSpan) [▶ 130]	Writes a time span as PLC time type to the current stream.
	WritePlcUnicodeString [▶ 131]	Writes a (unicode) string as a PLC string to the current stream.
	WritePlcUnicodeStringFixedLength [▶ 132]	Writes the PLC UNICODE string into a data block of the specified size.

Reference

AdsBinaryWriter Class [▶ 121]

TwinCAT.Ads Namespace [▶ 108]

5.2.2.3.1 AdsBinaryWriter.WriteGuid Method

Writes the Guid (16 Bytes) to the current stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteGuid(
    Guid guid
)
```

VB

```
Public Sub WriteGuid (
    guid As Guid
)
```

Parameters

guid	Type: System.Guid The unique identifier.

Reference

AdsBinaryWriter Class [▶ 121]

TwinCAT.Ads Namespace [▶ 108]

5.2.2.3.2 AdsBinaryWriter.WritePlcAnsiString Method

Writes a string as a PLC string to the current stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WritePlcAnsiString(
    string value,
    int length
)
```

VB

```
Public Sub WritePlcAnsiString (
    value As String,
    length As Integer
)
```

Parameters

value	Type: System.String The string to write to the stream.
length	Type: System.Int32 The length of the string without '\0' terminator!

Remarks

This method is meant for writing single string variables defined in the PlcControl format. E.g. to write a 'STRING(80)' (byte size is 81) a length of '80' must be given to the 'length' parameter. If the string length is larger or equal than the length parameter, then only length characters are written to the AdsStream [▶ 375] (without terminating character). If the string value character count is shorter than the specified length parameter, the string + a terminating \0 will be added to the AdsStream [▶ 375]. This method cannot be used for marshalling purposes, for example several fields of a struct, because no filling bytes will be written to the stream. In that case use the WritePlcAnsiStringFixedLength(String, Int32) [▶ 129] method.

Examples

The following code shows how to Read/Write ANSI string values..

Read/Write ANSI Strings

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local port 851 (PLC)

    int handle = client.CreateVariableHandle("MAIN.string"); // Symbol "string" in MAIN defined as s
tring

    try
    {
    // Read ANSI String string[80]
    int byteSize = 81; // Size of 80 ANSI chars + /0 (STRING[80])
    AdsStream readStream = new AdsStream(byteSize); // Size of 80 ANSI chars + /0 (STRING[80])
    AdsBinaryReader reader = new AdsBinaryReader(readStream);
    client.Read(handle, readStream, 0, byteSize); // Read 81 bytes
    string value = reader.ReadPlcString(byteSize,Encoding.Default);

    // Write ANSI String string[80]
    AdsStream writeStream = new AdsStream(byteSize);
    AdsBinaryWriter writer = new AdsBinaryWriter(writeStream);
    value = "Changed";
    writer.WritePlcString(value, 80,Encoding.Default); // Max 80 characters!
    client.Write(handle, writeStream, 0, byteSize);
    }
    finally
    {
    client.DeleteVariableHandle(handle);
    }
}
```

Reference

AdsBinaryWriter Class [▶ 121]

TwinCAT.Ads Namespace [▶ 108]

AdsBinaryWriter.WritePlcAnsiStringFixedLength(String, Int32) [▶ 129]

AdsBinaryWriter.WritePlcUnicodeString(String, Int32) [▶ 131]

AdsBinaryWriter.WritePlcUnicodeStringFixedLength(String, Int32) [▶ 132]

5.2.2.3.3 AdsBinaryWriter.WritePlcAnsiStringFixedLength Method

Writes the PLC ANSI string in a data block of the specified size.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WritePlcAnsiStringFixedLength(
    string value,
    int byteSize
)
```

VB

```
Public Sub WritePlcAnsiStringFixedLength (
    value As String,
    byteSize As Integer
)
```

Parameters

value	Type: System.String The value.
byteSize	Type: System.Int32 Size of the String including the '\0' terminator.

Reference

AdsBinaryWriter Class [▶ 121]

TwinCAT.Ads Namespace [▶ 108]

AdsBinaryWriter.WritePlcAnsiString(String, Int32) [▶ 127]

AdsBinaryWriter.WritePlcUnicodeString(String, Int32) [▶ 131]

AdsBinaryWriter.WritePlcUnicodeStringFixedLength(String, Int32) [▶ 132]

5.2.2.3.4 AdsBinaryWriter.WritePlcType Method

Overload List

	Name	Description
	WritePlcType(DateTime) [▶ 130]	Writes a date as PLC date type to the current stream.
	WritePlcType(TimeSpan) [▶ 130]	Writes a time span as PLC time type to the current stream.

Reference

AdsBinaryWriter Class [▶ 121]

TwinCAT.Ads Namespace [▶ 108]

AdsBinaryWriter.WritePlcType Method (DateTime)

Writes a date as PLC date type to the current stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WritePlcType(
    DateTime value
)
```

VB

```
Public Sub WritePlcType (
    value As DateTime
)
```

Parameters

value	Type: System.DateTime The date to write to the stream.

Reference

AdsBinaryWriter Class [▶ 121]

WritePlcType Overload [▶ 130]

TwinCAT.Ads Namespace [▶ 108]

AdsBinaryWriter.WritePlcType Method (TimeSpan)

Writes a time span as PLC time type to the current stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WritePlcType(
    TimeSpan value
)
```

VB

```
Public Sub WritePlcType (
    value As TimeSpan
)
```

Parameters

value	Type: System.TimeSpan The time span to write to the stream.

Reference

AdsBinaryWriter Class [▶ 121]

WritePlcType Overload [▶ 130]

TwinCAT.Ads Namespace [▶ 108]

5.2.2.3.5 AdsBinaryWriter.WritePlcUnicodeString Method

Writes a (unicode) string as a PLC string to the current stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WritePlcUnicodeString(
    string value,
    int length
)
```

VB

```
Public Sub WritePlcUnicodeString (
    value As String,
    length As Integer
)
```

Parameters

value	Type: System.String The string to write to the stream.
length	Type: System.Int32 The length of the string without '\0' terminator!

Remarks

This method is meant for writing single string variables defined in the PlcControl format. E.g. to write a 'WSTRING(80)' (byte size is 162) a length of '80' must be given to the 'length' parameter. If the string length is larger or equal than the length parameter, then only length characters are written to the AdsStream [▶ 375] (without terminating character). If the string value character count is shorter than the specified length

parameter, the string + a terminating \0 will be added to the AdsStream [▶ 375]. This method cannot be used for marshalling purposes, for example several fields of a struct, because no filling bytes will be written to the stream. In that case use the WritePlcUnicodeStringFixedLength(String, Int32) [▶ 132] method.

Examples

The following code shows how to Read/Write UNICODE string values..

Read/Write Unicode Strings

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local port 851 (PLC)

    int handle = client.CreateVariableHandle("MAIN.wstring"); // Symbol "wstring" defined in MAIN as
 WSTRING

    try
    {
    // Read UNICODE String wstring[80]
    int byteSize = 2 * 81; // Size of 80 UNICODE chars + /0 (WSTRING[80])
    AdsStream readStream = new AdsStream(byteSize);
    AdsBinaryReader reader = new AdsBinaryReader(readStream);
    client.Read(handle, readStream, 0, byteSize); // Read 2*81 bytes
    string value = reader.ReadPlcString(byteSize,Encoding.Unicode);

    // Write ANSI String string[80]
    AdsStream writeStream = new AdsStream(byteSize);
    AdsBinaryWriter writer = new AdsBinaryWriter(writeStream);
    value = "Changed";
    writer.WritePlcString(value, 80,Encoding.Unicode);
    client.Write(handle, writeStream, 0, byteSize);
    }
    finally
    {
    client.DeleteVariableHandle(handle);
    }
}
```

Reference

AdsBinaryWriter Class [▶ 121]

TwinCAT.Ads Namespace [▶ 108]

AdsBinaryWriter.WritePlcAnsiString(String, Int32) [▶ 127]

AdsBinaryWriter.WritePlcAnsiStringFixedLength(String, Int32) [▶ 129]

AdsBinaryWriter.WritePlcUnicodeStringFixedLength(String, Int32) [▶ 132]

5.2.2.3.6 AdsBinaryWriter.WritePlcUnicodeStringFixedLength Method

Writes the PLC UNICODE string into a data block of the specified size.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WritePlcUnicodeStringFixedLength(
    string value,
    int byteSize
)
```

VB

```
Public Sub WritePlcUnicodeStringFixedLength (
    value As String,
    byteSize As Integer
)
```

Parameters

value	Type: System.String The value.
byteSize	Type: System.Int32 Size of the String including the '\0' terminator.

Reference

AdsBinaryWriter Class [▶ 121]

TwinCAT.Ads Namespace [▶ 108]

AdsBinaryWriter.WritePlcAnsiStringFixedLength(String, Int32) [▶ 129]

AdsBinaryWriter.WritePlcUnicodeString(String, Int32) [▶ 131]

AdsBinaryWriter.WritePlcAnsiString(String, Int32) [▶ 127]

5.2.2.4 AdsBinaryWriter Fields

The AdsBinaryWriter [▶ 121] type exposes the following members.

Fields

	Name	Description
	OutStream	Holds the underlying stream. (Inherited from BinaryWriter.)

Reference

AdsBinaryWriter Class [▶ 121]

TwinCAT.Ads Namespace [▶ 108]

5.2.3 AdsClientSettings Class

Settings object for TcAdsClient [▶ 625].

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.AdsClientSettings
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class AdsClientSettings
```

VB

```
Public Class AdsClientSettings
```

The AdsClientSettings type exposes the following members.

Constructors

	Name	Description
	AdsClientSettings [▶ 135]	Creates a Default settings AdsClientSettings object with custom timeout.

Properties

	Name	Description
	CompatibilityDefault [▶ 136]	Compatibility settings object
	Default [▶ 136]	Gets the default settings.
	FastWriteThrough [▶ 137]	Gets a Settings object that configures the TcAdsClient for FastWriteThrough
	Protocol [▶ 138]	Gets the protocol settings
	Synchronize [▶ 138]	Gets a value indicating whether the TwinCAT.Ads [▶ 108]
	Timeout [▶ 138]	The communication Timeout of the TcAdsClient [▶ 625]

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Remarks

This AdsClientSettings object is used to initalize the TcAdsClient [▶ 625] with application appropriate settings.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.3.1 AdsClientSettings Constructor

Creates a Default settings AdsClientSettings [▶ 133] object with custom timeout.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsClientSettings(
    int timeout
)
```

VB

```
Public Sub New (
    timeout As Integer
)
```

Parameters

timeout	Type: System.Int32 The timeout of the TcAdsClient [▶ 625] in milliseconds.

Reference

AdsClientSettings Class [▶ 133]

TwinCAT.Ads Namespace [▶ 108]

5.2.3.2 AdsClientSettings Properties

The AdsClientSettings [▶ 133] type exposes the following members.

Properties

	Name	Description
	CompatibilityDefault [▶ 136]	Compatibility settings object
	Default [▶ 136]	Gets the default settings.
	FastWriteThrough [▶ 137]	Gets a Settings object that configures the TcAdsClient for FastWriteThrough
	Protocol [▶ 138]	Gets the protocol settings
	Synchronize [▶ 138]	Gets a value indicating whether the TwinCAT.Ads [▶ 108]
	Timeout [▶ 138]	The communication Timeout of the TcAdsClient [▶ 625]

Reference

AdsClientSettings Class [▶ 133]

TwinCAT.Ads Namespace [▶ 108]

5.2.3.2.1 AdsClientSettings.CompatibilityDefault Property

Compatibility settings object

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static AdsClientSettings CompatibilityDefault { get; }
```

VB

```
Public Shared ReadOnly Property CompatibilityDefault As AdsClientSettings
    Get
```

Property Value

Type: AdsClientSettings [▶ 133]
The settings object.

Remarks

The compatibility settings intitialize the TcAdsClient the same way as it is done in earlier versions of the TwinCAT.Ads.dll (earlier than Version 4.2)

- All [▶ 820]
- No FailFastHandler [▶ 523] active.
- Default communicationtimeout 5000ms.
- Synchronized Notifications.

Reference

AdsClientSettings Class [▶ 133]

TwinCAT.Ads Namespace [▶ 108]

5.2.3.2.2 AdsClientSettings.Default Property

Gets the default settings.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static AdsClientSettings Default { get; }
```

VB

```
Public Shared ReadOnly Property Default As AdsClientSettings
    Get
```

Property Value

Type: AdsClientSettings [▶ 133]
The default.

Remarks

Creates an settings object, with specification for All [▶ 820] and IFailFastHandler [▶ 523].

- All [▶ 820]
- Fail fast handler [▶ 523] is active.
- Default communication timeout 5000ms.
- Not synchronized Notifications.

Reference

AdsClientSettings Class [▶ 133]

TwinCAT.Ads Namespace [▶ 108]

5.2.3.2.3 AdsClientSettings.FastWriteThrough Property

Gets a Settings object that configures the TcAdsClient for FastWriteThrough

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static AdsClientSettings FastWriteThrough { get; }
```

VB

```
Public Shared ReadOnly Property FastWriteThrough As AdsClientSettings
    Get
```

Property Value

Type: AdsClientSettings [▶ 133]
Client settings for a fast write through (with 200 ms Timeout).

Remarks

The settings typically can be used for polling clients, where the "FailFast" feature will be bypassed. That means, that communication fails doesn't trigger the FailFast interceptor and every Request will go out via ADS. This has the Drawback that communication Timeouts are longer and subsequent timeouts block the ADS mailbox (with the danger of overflows). So use this setting with care for specific purposes and should not be used for standard communication.

- No FailFastHandler [▶ 523] active.
- Default communicationtimeout 200ms.
- Not synchronized Notifications.

Reference

AdsClientSettings Class [▶ 133]

TwinCAT.Ads Namespace [▶ 108]

5.2.3.2.4 AdsClientSettings.Protocol Property

Gets the protocol settings

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TransportProtocol Protocol { get; }
```

VB

```
Public ReadOnly Property Protocol As TransportProtocol
    Get
```

Property Value

Type: TransportProtocol [▶ 820]
The protocol.

Reference

AdsClientSettings Class [▶ 133]

TwinCAT.Ads Namespace [▶ 108]

5.2.3.2.5 AdsClientSettings.Synchronize Property

Gets a value indicating whether the TwinCAT.Ads [▶ 108]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Synchronize { get; }
```

VB

```
Public ReadOnly Property Synchronize As Boolean
    Get
```

Property Value

Type: Boolean
true if synchronize; otherwise, false.

Reference

AdsClientSettings Class [▶ 133]

TwinCAT.Ads Namespace [▶ 108]

5.2.3.2.6 AdsClientSettings.Timeout Property

The communication Timeout of the TcAdsClient [▶ 625]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Timeout { get; }
```

VB

```
Public ReadOnly Property Timeout As Integer
    Get
```

Property Value

Type: Int32
The timeout.

Reference

AdsClientSettings Class [▶ 133]

TwinCAT.Ads Namespace [▶ 108]

5.2.3.3 AdsClientSettings Methods

The AdsClientSettings [▶ 133] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

AdsClientSettings Class [▶ 133]

TwinCAT.Ads Namespace [▶ 108]

5.2.4 AdsCommunicationStatistics Class

ADS Communication statistics

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.AdsCommunicationStatistics
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class AdsCommunicationStatistics
```

VB

```
Public Class AdsCommunicationStatistics
```

The AdsCommunicationStatistics type exposes the following members.

Properties

	Name	Description
	AccessWaitTime [▶ 142]	Gets the wait time for the next access (Resurrection time) if in Lost [▶ 39].
	ConnectionActiveSince [▶ 142]	Gets the UTC time of the last conenction activation.
	ConnectionEstablishedAt [▶ 143]	Gets the UTC time when the current connection was established.
	ConnectionLostCount [▶ 143]	Gets the connection lost count.
	ConnectionLostTime [▶ 144]	Gets the UTC connection lost time.
	ConnectionResurrections [▶ 144]	Gets the number of resurrections on the AdsConnection [▶ 148]
	ErrorsSinceLastSucceeded [▶ 145]	Gets the error count since last access (UTC)
	LastSucceededAccess [▶ 145]	Gets the UTC time of the last succeeded access.
	Resurrections [▶ 146]	Gets the number of Resurrections of this Session.
	SessionEstablishedAt [▶ 146]	Gets the UTC time when the session was established.
	TotalCycles [▶ 147]	Gets the total cycles.
	TotalErrors [▶ 147]	Gets the total error count.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Remarks

These statistics can be used for communication diagnosis. They contain Error/Succeed counts as well as Resurrection infos.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1 AdsCommunicationStatistics Properties

The AdsCommunicationStatistics [▶ 139] type exposes the following members.

Properties

	Name	Description
	AccessWaitTime [▶ 142]	Gets the wait time for the next access (Resurrection time) if in Lost [▶ 39].
	ConnectionActiveSince [▶ 142]	Gets the UTC time of the last conenction activation.
	ConnectionEstablishedAt [▶ 143]	Gets the UTC time when the current connection was established.
	ConnectionLostCount [▶ 143]	Gets the connection lost count.
	ConnectionLostTime [▶ 144]	Gets the UTC connection lost time.
	ConnectionResurrections [▶ 144]	Gets the number of resurrections on the AdsConnection [▶ 148]
	ErrorsSinceLastSucceeded [▶ 145]	Gets the error count since last access (UTC)
	LastSucceededAccess [▶ 145]	Gets the UTC time of the last succeeded access.

	Name	Description
	Resurrections [▶ 146]	Gets the number of Resurrections of this Session.
	SessionEstablishedAt [▶ 146]	Gets the UTC time when the session was established.
	TotalCycles [▶ 147]	Gets the total cycles.
	TotalErrors [▶ 147]	Gets the total error count.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1.1 AdsCommunicationStatistics.AccessWaitTime Property

Gets the wait time for the next access (Resurrection time) if in Lost [▶ 39].

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TimeSpan AccessWaitTime { get; }
```

VB

```
Public ReadOnly Property AccessWaitTime As TimeSpan
    Get
```

Property Value

Type: TimeSpan
The wait time if in Lost [▶ 39] otherwise **TimeSpan.Zero**.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1.2 AdsCommunicationStatistics.ConnectionActiveSince Property

Gets the UTC time of the last conenction activation.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Nullable<DateTime> ConnectionActiveSince { get; }
```

VB

```
Public ReadOnly Property ConnectionActiveSince As Nullable(Of DateTime)
    Get
```

Property Value

Type: Nullable.DateTime.
Connection active time.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1.3 AdsCommunicationStatistics.ConnectionEstablishedAt Property

Gets the UTC time when the current connection was established.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Nullable<DateTime> ConnectionEstablishedAt { get; }
```

VB

```
Public ReadOnly Property ConnectionEstablishedAt As Nullable(Of DateTime)
    Get
```

Property Value

Type: Nullable.DateTime.
The connection established at.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1.4 AdsCommunicationStatistics.ConnectionLostCount Property

Gets the connection lost count.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ConnectionLostCount { get; }
```

VB

```
Public ReadOnly Property ConnectionLostCount As Integer
    Get
```

Property Value

Type: Int32
The connection lost count.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1.5 AdsCommunicationStatistics.ConnectionLostTime Property

Gets the UTC connection lost time.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Nullable<DateTime> ConnectionLostTime { get; }
```

VB

```
Public ReadOnly Property ConnectionLostTime As Nullable(Of DateTime)
    Get
```

Property Value

Type: Nullable.DateTime.
The connection lost time.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1.6 AdsCommunicationStatistics.ConnectionResurrections Property

Gets the number of resurrections on the AdsConnection [▶ 148]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ConnectionResurrections { get; }
```

VB

```
Public ReadOnly Property ConnectionResurrections As Integer
    Get
```

Property Value

Type: Int32
The resurrections.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1.7 AdsCommunicationStatistics.ErrorsSinceLastSucceeded Property

Gets the error count since last access (UTC)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ErrorsSinceLastSucceeded { get; }
```

VB

```
Public ReadOnly Property ErrorsSinceLastSucceeded As Integer
    Get
```

Property Value

Type: Int32
The error count since last access.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1.8 AdsCommunicationStatistics.LastSucceededAccess Property

Gets the UTC time of the last succeeded access.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Nullable<DateTime> LastSucceededAccess { get; }
```

VB

```
Public ReadOnly Property LastSucceededAccess As Nullable(Of DateTime)
    Get
```

Property Value

Type: Nullable.DateTime.
The last succeeded access.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1.9 AdsCommunicationStatistics.Resurrections Property

Gets the number of Resurrections of this Session.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Resurrections { get; }
```

VB

```
Public ReadOnly Property Resurrections As Integer
    Get
```

Property Value

Type: Int32
The resurrections.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1.10 AdsCommunicationStatistics.SessionEstablishedAt Property

Gets the UTC time when the session was established.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DateTime SessionEstablishedAt { get; }
```

VB

```
Public ReadOnly Property SessionEstablishedAt As DateTime
    Get
```

Property Value

Type: DateTime
The session established at.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1.11 AdsCommunicationStatistics.TotalCycles Property

Gets the total cycles.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int TotalCycles { get; }
```

VB

```
Public ReadOnly Property TotalCycles As Integer
    Get
```

Property Value

Type: Int32
The total cycles.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.1.12 AdsCommunicationStatistics.TotalErrors Property

Gets the total error count.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int TotalErrors { get; }
```

VB

```
Public ReadOnly Property TotalErrors As Integer
    Get
```

Property Value

Type: Int32
The total error count.

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.4.2 AdsCommunicationStatistics Methods

The AdsCommunicationStatistics [▶ 139] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

AdsCommunicationStatistics Class [▶ 139]

TwinCAT.Ads Namespace [▶ 108]

5.2.5 AdsConnection Class

ADS Connection class

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.AdsConnection
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class AdsConnection : IAdsConnection,
    IConnection, IConnectionStateProvider, IAdsNotifications, IAdsAnyAccess, IAdsHandleAccess,
    ITcAdsRpcInvoke, IDisposable
```

VB

```
Public NotInheritable Class AdsConnection
    Implements IAdsConnection, IConnection, IConnectionStateProvider, IAdsNotifications,
    IAdsAnyAccess, IAdsHandleAccess, ITcAdsRpcInvoke, IDisposable
```

The AdsConnection type exposes the following members.

Properties

	Name	Description
	AccessWaitTime [▶ 161]	Gets the access wait time.
	ActiveSince [▶ 162]	Gets the UTC time when tha last active/resurrected Connection was established
	Address [▶ 162]	Gets the AmsAddress [▶ 410] of the ADS server.
	ClientAddress [▶ 163]	Get the AmsAddress [▶ 410] of the ADS client.
	ConnectionEstablishedAt [▶ 163]	Gets the UTC time when the Connection was originally established.
	ConnectionLostCount [▶ 164]	Gets the connection lost count.
	ConnectionLostTime [▶ 164]	Gets the connection lost time.
	ConnectionState [▶ 165]	Gets the current Connection state of the AdsConnection
	Disposed [▶ 166]	Gets a value indicating whether this AdsConnection is disposed.
	Id [▶ 167]	Gets the AdsConnection identifier.
	IsActive [▶ 167]	Gets a value indicating whether communication is in active state
	IsConnected [▶ 168]	Gets a value indicating whether the local ADS port was opened successfully. It does not indicate if the target port is available. Use the method ReadState to determine if the target port is available.
	IsLocal [▶ 168]	Gets a value indicating whether the ADS client is connected to a ADS Server on the local computer.
	IsLost [▶ 169]	Gets a value indicating whether the communication is in lost / open state
	IsReconnecting [▶ 169]	Gets a value indicating whether communication is ready for reconnecting
	Name [▶ 170]	Gets the name.
	ResurrectingTries [▶ 170]	Gets the number of tries to resurrect the AdsConnection.
	Resurrections [▶ 171]	Gets the number of succeeded connection resurrections.
	Session [▶ 171]	Gets the Session object of the AdsConnection object.
	State [▶ 172]	Gets the current ConnectionState [▶ 165]

	Name	Description
	Timeout [▶ 172]	Gets the timeout (in milliseconds)

Methods

	Name	Description
	AddDeviceNotification(String, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 183]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 184]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 186]	Adds the device notification.
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 187]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type) [▶ 189]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 190]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32,	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AdsTransMode, Int32, Int32, Object, Type) [▶ 191]	
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 192]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	Close [▶ 194]	Closes the AdsConnection
	Connect [▶ 194]	(Re)Connects the IConnection [▶ 46] when disconnected.
	CreateSymbolLoader [▶ 195]	Creates a new instance of the Symbol loader [▶ 979] with the specified mode.
	CreateVariableHandle [▶ 206]	Generates a unique handle for an ADS variable.
	DeleteDeviceNotification [▶ 207]	Deletes an existing notification.
	DeleteVariableHandle [▶ 207]	Releases the handle of a ADS variable again.
	Disconnect [▶ 208]	Disconnects this IConnection [▶ 46].
	Dispose [▶ 208]	Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	InvokeRpcMethod(String, Int32, .Object.) [▶ 210]	Invokes the specified RPC Method
	InvokeRpcMethod(String, String, .Object.) [▶ 210]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 211]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 212]	Invokes the specified RPC Method.

	Name	Description
	Read(Int32, AdsStream) [▶ 214]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, AdsStream) [▶ 215]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(Int32, AdsStream, Int32, Int32) [▶ 216]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, .Byte., Int32, Int32) [▶ 216]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 217]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, .Byte., Int32, Int32, Int32) [▶ 218]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, AdsStream, Int32, Int32, Int32) [▶ 219]	Reads data synchronously from an ADS device and writes it to the given stream.
	ReadAny(Int32, Type) [▶ 221]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(Int32, Type, .Int32.) [▶ 222]	Reads any.
	ReadAny(UInt32, UInt32, Type) [▶ 223]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 223]	Reads data synchronously from an ADS device and writes it to an object. If the Type of the object to be read is a string type, the first element of the parameter args specifies the number of characters of the string. If the Type of the object to be read is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	ReadAny(UInt32, UInt32, Type, .Int32., Int32) [▶ 224]	Reads data synchronously from an ADS device and writes it to an object. If the Type of the object to be read is a string type, the first element of the parameter args specifies the number of characters of the string. If the Type of the object to be read is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	ReadAnyString(Int32, Int32, Encoding) [▶ 226]	Reads the string.
	ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 227]	Reads the string
	ReadDeviceInfo [▶ 228]	Reads the identification and version number of an ADS server.
	ReadState. [▶ 229]	Reads the ADS status and the device status from an ADS server.

	Name	Description
	ReadState(Int32) [▶ 229]	Reads the ADS status and the device status from an ADS server.
	ReadSymbol(ITcAdsSymbol) [▶ 230]	Reads the value of a symbol and returns it as an object. Strings and all primitive datatypes(UInt32, Int32, Bool etc.) are supported. Arrays and structures cannot be read.
	ReadSymbol(String, Type, Boolean) [▶ 231]	Reads the value of a symbol and returns the value as object. The parameter type must have the same layout as the ADS symbol.
	ReadSymbolInfo [▶ 232]	Call this method to obtain information about the individual symbols (variables) in ADS devices.
	ReadWrite(UInt32, UInt32, AdsStream, AdsStream) [▶ 233]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, AdsStream, AdsStream, Int32) [▶ 234]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 235]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32) [▶ 236]	Writes data synchronously to an ADS device and then reads data from this device.
	ReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 237]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32) [▶ 239]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32) [▶ 240]	Writes data synchronously to an ADS device and then Reads data from this device.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryAddDeviceNotification [▶ 241]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	TryAddDeviceNotificationEx [▶ 242]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	TryDeleteDeviceNotification [▶ 243]	Deletes an existing notification.

	Name	Description
	TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 245]	Invokes the specified RPC Method
	TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 246]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 247]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 248]	Invokes the specified RPC Method.
	TryRead(Int32, AdsStream, Int32, Int32, Int32.) [▶ 249]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, .Byte., Int32, Int32, Int32.) [▶ 250]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, AdsStream, Int32, Int32, Int32.) [▶ 251]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, .Byte., Int32, Int32, Int32, Int32.) [▶ 252]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, AdsStream, Int32, Int32, Int32, Int32.) [▶ 253]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryReadState(StateInfo.) [▶ 255]	Reads the ADS status and the device status from an ADS server. Unlike the ReadState method this method does not call an exception on failure. Instead an AdsErrorCode is returned. If the return value is equal to AdsErrorCode.NoError the call was successful.
	TryReadState(Int32, StateInfo.) [▶ 255]	Reads the ADS status and the device status from an ADS server. Unlike the ReadState method this method does not call an exception on failure. Instead an AdsErrorCode is returned. If the return value is equal to AdsErrorCode.NoError the call was successfull.
	TryReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 257]	Writes data synchronously to an ADS device and then Reads data from this device.

	Name	Description
	TryReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32.) [▶ 258]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 260]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32, Int32.) [▶ 261]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32, Int32.) [▶ 262]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryWrite(Int32, AdsStream, Int32, Int32) [▶ 264]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, .Byte., Int32, Int32) [▶ 265]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 266]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, .Byte., Int32, Int32, Int32) [▶ 267]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, AdsStream, Int32, Int32, Int32) [▶ 268]	Writes data synchronously to an ADS device.
	TryWriteControl(StateInfo) [▶ 269]	Changes the ADS status and the device status of an ADS server.
	TryWriteControl(StateInfo, Int32) [▶ 270]	Changes the ADS status and the device status of an ADS server.

	Name	Description
	TryWriteControl(StateInfo, AdsStream, Int32, Int32) [▶ 271]	Changes the ADS status and the device status of an ADS server.
	TryWriteControl(StateInfo, AdsStream, Int32, Int32, Int32) [▶ 272]	Changes the ADS status and the device status of an ADS server.
	Write(Int32, AdsStream) [▶ 274]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32) [▶ 274]	Trigger Client Method/Command.
	Write(UInt32, UInt32, Int32) [▶ 275]	Trigger Client Method/Command.
	Write(UInt32, UInt32, AdsStream) [▶ 276]	Writes data synchronously to an ADS device.
	Write(Int32, AdsStream, Int32, Int32) [▶ 276]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream, Int32) [▶ 277]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, .Byte., Int32, Int32) [▶ 278]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 279]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, .Byte., Int32, Int32, Int32) [▶ 280]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream, Int32, Int32, Int32) [▶ 281]	Writes data synchronously to an ADS device.
	WriteAny(Int32, Object) [▶ 282]	Writes an object synchronously to an ADS device.
	WriteAny(Int32, Object, .Int32.) [▶ 283]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.
	WriteAny(UInt32, UInt32, Object) [▶ 283]	Writes an object synchronously to an ADS device.
	WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 284]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.

	Name	Description
	WriteAny(UInt32, UInt32, Object, .Int32., Int32) [▶ 285]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.
	WriteControl(StateInfo) [▶ 287]	Changes the ADS status and the device status of an ADS server.
	WriteControl(StateInfo, Int32) [▶ 287]	Changes the ADS status and the device status of an ADS server.
	WriteControl(StateInfo, AdsStream, Int32, Int32) [▶ 288]	Changes the ADS status and the device status of an ADS server.
	WriteControl(StateInfo, AdsStream, Int32, Int32, Int32) [▶ 289]	Changes the ADS status and the device status of an ADS server.
	WriteSymbol(ITcAdsSymbol, Object) [▶ 290]	Writes a value to the symbol. Strings and all primitive data types(UInt32, Int32, Bool etc.) are supported. Array and structures are not supported. If a string is passed as parameter, the method attempts to parse the string according to the ADS data type of the symbol.
	WriteSymbol(String, Object, Boolean) [▶ 291]	Writes the passed object value to the specified ADS symbol.The parameter type must have the same layout as the ADS symbol.

Events

	Name	Description
	AdsNotification [▶ 292]	Occurs when the ADS device sends a notification to the client.
	AdsNotificationError [▶ 292]	Occurs when a exception has occurred during notification management.
	AdsNotificationEx [▶ 293]	Occurs when the ADS devices sends a notification to the client.
	AdsStateChanged [▶ 293]	Occurs when ADS State has been changed.
	AdsSymbolVersionChanged [▶ 294]	Occurs when the symbol version has been changed.
	AmsRouterNotification [▶ 294]	Occurs when the Route sends an Notification.
	ConnectionStateChanged [▶ 295]	Occurs when connection status of the AdsConnection has been changed.

Extension Methods

	Name	Description
	PollAdsState(IObservable.Unit.) [▶ 825]	Overloaded. Gets an observable sequence of AdsState [▶ 365]s via Polling. (Defined by AdsClientExtensions [▶ 821].)
	PollAdsState(TimeSpan) [▶ 826]	Overloaded. Gets an observable sequence of AdsState [▶ 365]s via Polling. (Defined by AdsClientExtensions [▶ 821].)
	PollValues(String, Type, IObservable.Unit.) [▶ 849]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, TimeSpan) [▶ 850]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., TimeSpan) [▶ 853]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, IObservable.Unit., Func.Exception, Object.) [▶ 854]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, TimeSpan, Func.Exception, Object.) [▶ 855]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., IObservable.Unit., Func.Exception, Object.) [▶ 856]	Overloaded. Polls the symbol values on time points where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., TimeSpan, Func.Exception, Object.) [▶ 858]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, IObservable.Unit.) [▶ 842]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, TimeSpan) [▶ 843]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, IObservable.Unit., Func.Exception, T.) [▶ 847]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)

	Name	Description
	PollValues.T.(String, TimeSpan, Func.Exception, T.) [▶ 848]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T. (String, .Int32., IObservable.Unit.) [▶ 844]	Overloaded. Polls the symbol values on time points where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T. (String, .Int32., TimeSpan) [▶ 845]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T. (String, .Int32., IObservable.Unit., Func.Exception, T.) [▶ 851]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T. (String, .Int32., TimeSpan, Func.Exception, T.) [▶ 852]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	WhenNotification(ISymbol) [▶ 829]	Overloaded. Gets an observable sequence of Notification [▶ 862]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbolCollection) [▶ 830]	Overloaded. Gets an observable sequence of Notification [▶ 862] objects. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbol, NotificationSettings) [▶ 832]	Overloaded. Gets an observable sequence of Notification [▶ 862]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbolCollection, NotificationSettings) [▶ 833]	Overloaded. Gets an observable sequence of Notification [▶ 862] objects. (Defined by AdsClientExtensions [▶ 821].)
	WhenValueChanged [▶ 886]	Observable sequence of Values driven by ADS Notifications on the specified symbol. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues.T. (String, IObservable.T.) [▶ 859]	Overloaded. Writes the sequence of values to the symbol specified by the instance path. (Defined by AnyTypeExtensions [▶ 835].)
	WriteValues.T. (String, IObservable.T., Action.Exception.) [▶ 860]	Overloaded. Writes the sequence of values to the symbol specified by the instance path. (Defined by AnyTypeExtensions [▶ 835].)

Remarks

The ADS Connection class represents an ADS Point-to-Point Connection between client and server. It is established by using the Connect method of the AdsSession [▶ 347] object. An ADS Connection can have different ConnectionStates [▶ 165], which represent the state of the logical ADS connection.

Reference

TwinCAT.Ads Namespace [▶ 108]

TwinCAT.Ads.AdsSession [▶ 347]

TwinCAT.Ads.IAdsConnection [▶ 469]

System.IDisposable

5.2.5.1 AdsConnection Properties

The AdsConnection [▶ 148] type exposes the following members.

Properties

	Name	Description
	AccessWaitTime [▶ 161]	Gets the access wait time.
	ActiveSince [▶ 162]	Gets the UTC time when tha last active/resurrected Connection was established
	Address [▶ 162]	Gets the AmsAddress [▶ 410] of the ADS server.
	ClientAddress [▶ 163]	Get the AmsAddress [▶ 410] of the ADS client.
	ConnectionEstablishedAt [▶ 163]	Gets the UTC time when the Connection was originally established.
	ConnectionLostCount [▶ 164]	Gets the connection lost count.
	ConnectionLostTime [▶ 164]	Gets the connection lost time.
	ConnectionState [▶ 165]	Gets the current Connection state of the AdsConnection [▶ 148]
	Disposed [▶ 166]	Gets a value indicating whether this AdsConnection [▶ 148] is disposed.
	Id [▶ 167]	Gets the AdsConnection [▶ 148] identifier.
	IsActive [▶ 167]	Gets a value indicating whether communication is in active state
	IsConnected [▶ 168]	Gets a value indicating whether the local ADS port was opened successfully. It does not indicate if the target port is available. Use the method ReadState to determine if the target port is available.
	IsLocal [▶ 168]	Gets a value indicating whether the ADS client is connected to a ADS Server on the local computer.

	Name	Description
	IsLost [▶ 169]	Gets a value indicating whether the communication is in lost / open state
	IsReconnecting [▶ 169]	Gets a value indicating whether communication is ready for reconnecting
	Name [▶ 170]	Gets the name.
	ResurrectingTries [▶ 170]	Gets the number of tries to resurrect the AdsConnection [▶ 148].
	Resurrections [▶ 171]	Gets the number of succeeded connection resurrections.
	Session [▶ 171]	Gets the Session object of the AdsConnection [▶ 148] object.
	State [▶ 172]	Gets the current ConnectionState [▶ 165]
	Timeout [▶ 172]	Gets the timeout (in milliseconds)

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.1 AdsConnection.AccessWaitTime Property

Gets the access wait time.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TimeSpan AccessWaitTime { get; }
```

VB

```
Public ReadOnly Property AccessWaitTime As TimeSpan
    Get
```

Property Value

Type: TimeSpan
The access wait time.

Remarks

Gets the Wait Time until the next communication try will be done. This time is calculated as follows: ResurrectionTime - (DateTime.Now - ConnectionLostTime)

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ConnectionLostTime [▶ 164]

SessionSettings.ResurrectionTime [▶ 595]

5.2.5.1.2 AdsConnection.ActiveSince Property

Gets the UTC time when tha last active/resurrected Connection was established

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Nullable<DateTime> ActiveSince { get; }
```

VB

```
Public ReadOnly Property ActiveSince As Nullable(Of DateTime)
    Get
```

Property Value

Type: Nullable.DateTime.
The active since.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.3 AdsConnection.Address Property

Gets the AmsAddress [▶ 410] of the ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress Address { get; }
```

VB

```
Public ReadOnly Property Address As AmsAddress
    Get
```

Property Value

Type: AmsAddress [▶ 410]
The server address.

Implements

IAdsConnection.Address [▶ 477]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.4 AdsConnection.ClientAddress Property

Get the AmsAddress [▶ 410] of the ADS client.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress ClientAddress { get; }
```

VB

```
Public ReadOnly Property ClientAddress As AmsAddress
    Get
```

Property Value

Type: AmsAddress [▶ 410]
The client address.

Implements

IAdsConnection.ClientAddress [▶ 477]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.5 AdsConnection.ConnectionEstablishedAt Property

Gets the UTC time when the Connection was originally established.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Nullable<DateTime> ConnectionEstablishedAt { get; }
```

VB

```
Public ReadOnly Property ConnectionEstablishedAt As Nullable(Of DateTime)
    Get
```

Property Value

Type: Nullable.DateTime.
The connection established at.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.6 AdsConnection.ConnectionLostCount Property

Gets the connection lost count.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ConnectionLostCount { get; }
```

VB

```
Public ReadOnly Property ConnectionLostCount As Integer
    Get
```

Property Value

Type: Int32
The connection lost count.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.7 AdsConnection.ConnectionLostTime Property

Gets the connection lost time.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Nullable<DateTime> ConnectionLostTime { get; }
```

VB

```
Public ReadOnly Property ConnectionLostTime As Nullable(Of DateTime)
    Get
```

Property Value

Type: Nullable.DateTime.
The connection lost time.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.8 AdsConnection.ConnectionState Property

Gets the current Connection state of the AdsConnection [▶ 148]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ConnectionState ConnectionState { get; }
```

VB

```
Public ReadOnly Property ConnectionState As ConnectionState
    Get
```

Property Value

Type: ConnectionState [▶ 39]
The state of the connection.

Implements

IConnectionStateProvider.ConnectionState [▶ 53]

Remarks

The Connection state changes only if the IConnection [▶ 46] is established / shut down or active communication is triggered by the User of the IConnection [▶ 46] object.

Examples

The following sample shows how to keep the ConnectionState updated by triggering ADS Communication.

Trigger ConnectionState changes in WPF Applications

```
private DispatcherTimer _timer = null;
private AdsSession _session = null;
//private AdsConnection _connection = null;

private void Window_Loaded(object sender, RoutedEventArgs e)
{
    _session = new AdsSession(AmsNetId.Local, 10000);
    IConnection connection = _session.Connect();
    tbConnectionState.Text = connection.ConnectionState.ToString();
    _session.ConnectionStateChanged += _session_ConnectionStateChanged;

    _timer = new DispatcherTimer();
    _timer.Interval = TimeSpan.FromMilliseconds(200);
    _timer.Tick += TimerOnTick;
```

```
    _timer.Start();
}

private void Window_Unloaded(object sender, RoutedEventArgs e)
{
    _timer.Stop();
    _session.Dispose();
}

private void _session_ConnectionStateChanged(object sender, TwinCAT.ConnectionStateChangedEventArgs
e)
{
    // ConnectionStateChanged will be triggered by communication Invokes
    tbConnectionState.Text = e.NewState.ToString();
}

private void TimerOnTick(object sender, EventArgs eventArgs)
{
    // The Timer Event will occur here in the UI thread because its an DispatcherTimer event!
    // An active ADS request will trigger Connection State periodically!
    StateInfo stateInfo;
    if (_session.Connection.TryReadState(out stateInfo) == AdsErrorCode.NoError)
    {
    tbAdsState.Text = stateInfo.AdsState.ToString();
    }
    else
    {
    tbAdsState.Text = "Invalid";
    }
}
```

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ConnectionStateChanged [▶ 295]

5.2.5.1.9 AdsConnection.Disposed Property

Gets a value indicating whether this AdsConnection [▶ 148] is disposed.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Disposed { get; }
```

VB

```
Public ReadOnly Property Disposed As Boolean
    Get
```

Property Value

Type: Boolean
true if disposed; otherwise, false.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.10 AdsConnection.Id Property

Gets the AdsConnection [▶ 148] identifier.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Id { get; }
```

VB

```
Public ReadOnly Property Id As Integer
    Get
```

Property Value

Type: Int32
The identifier.

Implements

IConnection.Id [▶ 48]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.11 AdsConnection.IsActive Property

Gets a value indicating whether communication is in active state

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsActive { get; }
```

VB

```
Public ReadOnly Property IsActive As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is active; otherwise, false.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.12 AdsConnection.IsConnected Property

Gets a value indicating whether the local ADS port was opened successfully. It does not indicate if the target port is available. Use the method ReadState to determine if the target port is available.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsConnected { get; }
```

VB

```
Public ReadOnly Property IsConnected As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is connected; otherwise, false.

Implements

IConnection.IsConnected [▶ 48]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.13 AdsConnection.IsLocal Property

Gets a value indicating whether the ADS client is connected to a ADS Server on the local computer.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsLocal { get; }
```

VB

```
Public ReadOnly Property IsLocal As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is local; otherwise, false.

Implements

IAdsConnection.IsLocal [▶ 477]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.14 AdsConnection.IsLost Property

Gets a value indicating whether the communication is in lost / open state

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsLost { get; }
```

VB

```
Public ReadOnly Property IsLost As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is lost; otherwise, false.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.15 AdsConnection.IsReconnecting Property

Gets a value indicating whether communication is ready for reconnecting

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReconnecting { get; }
```

VB

```
Public ReadOnly Property IsReconnecting As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is reconnecting; otherwise, false.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.16 AdsConnection.Name Property

Gets the name.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Name { get; }
```

VB

```
Public ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.17 AdsConnection.ResurrectingTries Property

Gets the number of tries to resurrect the AdsConnection [▶ 148].

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ResurrectingTries { get; }
```

VB

```
Public ReadOnly Property ResurrectingTries As Integer
    Get
```

Property Value

Type: Int32
The number of tried resurrections of the IConnection [▶ 46].

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.18 AdsConnection.Resurrections Property

Gets the number of succeeded connection resurrections.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Resurrections { get; }
```

VB

```
Public ReadOnly Property Resurrections As Integer
    Get
```

Property Value

Type: Int32
The resurrection count.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.19 AdsConnection.Session Property

Gets the Session object of the AdsConnection [▶ 148] object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ISession Session { get; }
```

VB

```
Public ReadOnly Property Session As ISession
    Get
```

Property Value

Type: ISession [▶ 60]
The client.

Implements

IConnection.Session [▶ 49]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.20 AdsConnection.State Property

Gets the current ConnectionState [▶ 165]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ConnectionState State { get; }
```

VB

```
Public ReadOnly Property State As ConnectionState
    Get
```

Property Value

Type: ConnectionState [▶ 39]
The state.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.1.21 AdsConnection.Timeout Property

Gets the timeout (in milliseconds)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Timeout { get; set; }
```

VB

```
Public Property Timeout As Integer
    Get
    Set
```

Property Value

Type: Int32
The timeout.

Implements

IConnection.Timeout [▶ 49]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2 AdsConnection Methods

The AdsConnection [▶ 148] type exposes the following members.

Methods

	Name	Description
	AddDeviceNotification(String, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 183]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 184]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 186]	Adds the device notification.
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 187]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type) [▶ 189]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 190]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type) [▶ 191]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 192]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	Close [▶ 194]	Closes the AdsConnection [▶ 148]
	Connect [▶ 194]	(Re)Connects the IConnection [▶ 46] when disconnected.
	CreateSymbolLoader [▶ 195]	Creates a new instance of the Symbol loader [▶ 979] with the specified mode.
	CreateVariableHandle [▶ 206]	Generates a unique handle for an ADS variable.
	DeleteDeviceNotification [▶ 207]	Deletes an existing notification.
	DeleteVariableHandle [▶ 207]	Releases the handle of a ADS variable again.
	Disconnect [▶ 208]	Disconnects this IConnection [▶ 46].
	Dispose [▶ 208]	Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	InvokeRpcMethod(String, Int32, .Object.) [▶ 210]	Invokes the specified RPC Method
	InvokeRpcMethod(String, String, .Object.) [▶ 210]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 211]	Invokes the specified RPC Method

	Name	Description
	InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 212]	Invokes the specified RPC Method.
	Read(Int32, AdsStream) [▶ 214]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, AdsStream) [▶ 215]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(Int32, AdsStream, Int32, Int32) [▶ 216]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, .Byte., Int32, Int32) [▶ 216]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 217]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, .Byte., Int32, Int32, Int32) [▶ 218]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, AdsStream, Int32, Int32, Int32) [▶ 219]	Reads data synchronously from an ADS device and writes it to the given stream.
	ReadAny(Int32, Type) [▶ 221]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(Int32, Type, .Int32.) [▶ 222]	Reads any.
	ReadAny(UInt32, UInt32, Type) [▶ 223]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 223]	Reads data synchronously from an ADS device and writes it to an object. If the Type of the object to be read is a string type, the first element of the parameter args specifies the number of characters of the string. If the Type of the object to be read is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	ReadAny(UInt32, UInt32, Type, .Int32., Int32) [▶ 224]	Reads data synchronously from an ADS device and writes it to an object. If the Type of the object to be read is a string type, the first element of the parameter args specifies the number of characters of the string. If the Type of the object to be read is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	ReadAnyString(Int32, Int32, Encoding) [▶ 226]	Reads the string.

	Name	Description
	ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 227]	Reads the string
	ReadDeviceInfo [▶ 228]	Reads the identification and version number of an ADS server.
	ReadState. [▶ 229]	Reads the ADS status and the device status from an ADS server.
	ReadState(Int32) [▶ 229]	Reads the ADS status and the device status from an ADS server.
	ReadSymbol(ITcAdsSymbol) [▶ 230]	Reads the value of a symbol and returns it as an object. Strings and all primitive datatypes(UInt32, Int32, Bool etc.) are supported. Arrays and structures cannot be read.
	ReadSymbol(String, Type, Boolean) [▶ 231]	Reads the value of a symbol and returns the value as object. The parameter type must have the same layout as the ADS symbol.
	ReadSymbolInfo [▶ 232]	Call this method to obtain information about the individual symbols (variables) in ADS devices.
	ReadWrite(UInt32, UInt32, AdsStream, AdsStream) [▶ 233]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, AdsStream, AdsStream, Int32) [▶ 234]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 235]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32) [▶ 236]	Writes data synchronously to an ADS device and then reads data from this device.
	ReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 237]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32) [▶ 239]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32) [▶ 240]	Writes data synchronously to an ADS device and then Reads data from this device.
	ToString	Returns a string that represents the current object. (Inherited from Object.)

	Name	Description
	TryAddDeviceNotification [▶ 241]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	TryAddDeviceNotificationEx [▶ 242]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	TryDeleteDeviceNotification [▶ 243]	Deletes an existing notification.
	TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 245]	Invokes the specified RPC Method
	TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 246]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 247]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 248]	Invokes the specified RPC Method.
	TryRead(Int32, AdsStream, Int32, Int32, Int32.) [▶ 249]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, .Byte., Int32, Int32, Int32.) [▶ 250]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, AdsStream, Int32, Int32, Int32.) [▶ 251]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, .Byte., Int32, Int32, Int32, Int32.) [▶ 252]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, AdsStream, Int32, Int32, Int32, Int32.) [▶ 253]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryReadState(StateInfo.) [▶ 255]	Reads the ADS status and the device status from an ADS server. Unlike the ReadState method this method does not call an exception on failure. Instead an AdsErrorCode is returned. If the return value is equal to AdsErrorCode.NoError the call was successful.

	Name	Description
	TryReadState(Int32, StateInfo.) [▶ 255]	Reads the ADS status and the device status from an ADS server. Unlike the ReadState method this method does not call an exception on failure. Instead an AdsErrorCode is returned. If the return value is equal to AdsErrorCode.NoError the call was successfull.
	TryReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 257]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32.) [▶ 258]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 260]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32, Int32.) [▶ 261]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32, Int32.) [▶ 262]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryWrite(Int32, AdsStream, Int32, Int32) [▶ 264]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, .Byte., Int32, Int32) [▶ 265]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 266]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, .Byte., Int32, Int32, Int32) [▶ 267]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, AdsStream, Int32, Int32, Int32) [▶ 268]	Writes data synchronously to an ADS device.

	Name	Description
	TryWriteControl(StateInfo) [▶ 269]	Changes the ADS status and the device status of an ADS server.
	TryWriteControl(StateInfo, Int32) [▶ 270]	Changes the ADS status and the device status of an ADS server.
	TryWriteControl(StateInfo, AdsStream, Int32, Int32) [▶ 271]	Changes the ADS status and the device status of an ADS server.
	TryWriteControl(StateInfo, AdsStream, Int32, Int32, Int32) [▶ 272]	Changes the ADS status and the device status of an ADS server.
	Write(Int32, AdsStream) [▶ 274]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32) [▶ 274]	Trigger Client Method/Command.
	Write(UInt32, UInt32, Int32) [▶ 275]	Trigger Client Method/Command.
	Write(UInt32, UInt32, AdsStream) [▶ 276]	Writes data synchronously to an ADS device.
	Write(Int32, AdsStream, Int32, Int32) [▶ 276]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream, Int32) [▶ 277]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, .Byte., Int32, Int32) [▶ 278]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 279]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, .Byte., Int32, Int32, Int32) [▶ 280]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream, Int32, Int32, Int32) [▶ 281]	Writes data synchronously to an ADS device.
	WriteAny(Int32, Object) [▶ 282]	Writes an object synchronously to an ADS device.
	WriteAny(Int32, Object, .Int32.) [▶ 283]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.

	Name	Description
	WriteAny(UInt32, UInt32, Object) [▶ 283]	Writes an object synchronously to an ADS device.
	WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 284]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.
	WriteAny(UInt32, UInt32, Object, .Int32., Int32) [▶ 285]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.
	WriteControl(StateInfo) [▶ 287]	Changes the ADS status and the device status of an ADS server.
	WriteControl(StateInfo, Int32) [▶ 287]	Changes the ADS status and the device status of an ADS server.
	WriteControl(StateInfo, AdsStream, Int32, Int32) [▶ 288]	Changes the ADS status and the device status of an ADS server.
	WriteControl(StateInfo, AdsStream, Int32, Int32, Int32) [▶ 289]	Changes the ADS status and the device status of an ADS server.
	WriteSymbol(ITcAdsSymbol, Object) [▶ 290]	Writes a value to the symbol. Strings and all primitive data types(UInt32, Int32, Bool etc.) are supported. Array and structures are not supported. If a string is passed as parameter, the method attempts to parse the string according to the ADS data type of the symbol.
	WriteSymbol(String, Object, Boolean) [▶ 291]	Writes the passed object value to the specified ADS symbol.The parameter type must have the same layout as the ADS symbol.

Extension Methods

	Name	Description
	PollAdsState(IObservable.Unit.) [▶ 825]	Overloaded. Gets an observable sequence of AdsState [▶ 365]s via Polling. (Defined by AdsClientExtensions [▶ 821].)
	PollAdsState(TimeSpan) [▶ 826]	Overloaded. Gets an observable sequence of AdsState [▶ 365]s via Polling. (Defined by AdsClientExtensions [▶ 821].)
	PollValues(String, Type, IObservable.Unit.) [▶ 849]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, TimeSpan) [▶ 850]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)

	Name	Description
	PollValues(String, Type, .Int32., TimeSpan) [▶ 853]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, IObservable.Unit., Func.Exception, Object.) [▶ 854]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, TimeSpan, Func.Exception, Object.) [▶ 855]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., IObservable.Unit., Func.Exception, Object.) [▶ 856]	Overloaded. Polls the symbol values on time points where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., TimeSpan, Func.Exception, Object.) [▶ 858]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, IObservable.Unit.) [▶ 842]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, TimeSpan) [▶ 843]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, IObservable.Unit., Func.Exception, T.) [▶ 847]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, TimeSpan, Func.Exception, T.) [▶ 848]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, .Int32., IObservable.Unit.) [▶ 844]	Overloaded. Polls the symbol values on time points where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, .Int32., TimeSpan) [▶ 845]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, .Int32., IObservable.Unit., Func.Exception, T.) [▶ 851]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)

	Name	Description
	PollValues.T. (String, .Int32., TimeSpan, Func.Exception, T.) [▶ 852]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	WhenNotification(ISymbol) [▶ 829]	Overloaded. Gets an observable sequence of Notification [▶ 862]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbolCollection) [▶ 830]	Overloaded. Gets an observable sequence of Notification [▶ 862] objects. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbol, NotificationSettings) [▶ 832]	Overloaded. Gets an observable sequence of Notification [▶ 862]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbolCollection, NotificationSettings) [▶ 833]	Overloaded. Gets an observable sequence of Notification [▶ 862] objects. (Defined by AdsClientExtensions [▶ 821].)
	WhenValueChanged [▶ 886]	Observable sequence of Values driven by ADS Notifications on the specified symbol. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues.T. (String, IObservable.T.) [▶ 859]	Overloaded. Writes the sequence of values to the symbol specified by the instance path. (Defined by AnyTypeExtensions [▶ 835].)
	WriteValues.T. (String, IObservable.T., Action.Exception.) [▶ 860]	Overloaded. Writes the sequence of values to the symbol specified by the instance path. (Defined by AnyTypeExtensions [▶ 835].)

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.1 AdsConnection.AddDeviceNotification Method

Overload List

	Name	Description
	AddDeviceNotification(String, AdsStream,	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AdsTransMode, Int32, Int32, Object) [▶ 183]	
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 184]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 186]	Adds the device notification.
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 187]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.AddDeviceNotification Method (String, AdsStream, AdsTransMode, Int32, Int32, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotification(
    string variableName,
    AdsStream dataStream,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData
)
```

VB

```
Public Function AddDeviceNotification (
    variableName As String,
    dataStream As AdsStream,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotification(String, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 500]

Reference

AdsConnection Class [▶ 148]

AddDeviceNotification Overload [▶ 182]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.AddDeviceNotification Method (UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotification(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData
)
```

VB

```
Public Function AddDeviceNotification (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 501]

Reference

AdsConnection Class [▶ 148]

AddDeviceNotification Overload [▶ 182]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.AddDeviceNotification Method (String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object)

Adds the device notification.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotification(
    string variableName,
    AdsStream dataStream,
    int offset,
    int length,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData
)
```

VB

```
Public Function AddDeviceNotification (
    variableName As String,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object
) As Integer
```

Parameters

variableName	Type: System.String Name of the variable.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The data stream.
offset	Type: System.Int32 The offset.
length	Type: System.Int32 The length.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] The trans mode.
cycleTime	Type: System.Int32 The cycle time.
maxDelay	Type: System.Int32 The maximum delay.
userData	Type: System.Object The user data.

Return Value

Type: Int32
System.Int32.

Implements

IAdsNotifications.AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 502]

Reference

AdsConnection Class [▶ 148]

AddDeviceNotification Overload [▶ 182]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.AddDeviceNotification Method (UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotification(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData
)
```

VB

```
Public Function AddDeviceNotification (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 504]

Reference

AdsConnection Class [▶ 148]

AddDeviceNotification Overload [▶ 182]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.2 AdsConnection.AddDeviceNotificationEx Method

Overload List

	Name	Description
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type) [▶ 189]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 190]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type) [▶ 191]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 192]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.AddDeviceNotificationEx Method (String, AdsTransMode, Int32, Int32, Object, Type)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotificationEx(
    string variableName,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData,
    Type type
)
```

VB

```
Public Function AddDeviceNotificationEx (
    variableName As String,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object,
    type As Type
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

type	Type: System.Type Type of the object stored in the event argument.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type) [▶ 506]

Reference

AdsConnection Class [▶ 148]

AddDeviceNotificationEx Overload [▶ 188]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.AddDeviceNotificationEx Method (String, AdsTransMode, Int32, Int32, Object, Type, .Int32.)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotificationEx(
    string variableName,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData,
    Type type,
    int[] args
)
```

VB

```
Public Function AddDeviceNotificationEx (
    variableName As String,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object,
    type As Type,
    args As Integer()
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.

cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 507]

Reference

AdsConnection Class [▶ 148]

AddDeviceNotificationEx Overload [▶ 188]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.AddDeviceNotificationEx Method (UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotificationEx(
    uint indexGroup,
    uint indexOffset,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData,
    Type type
)
```

VB

```
Public Function AddDeviceNotificationEx (
    indexGroup As UInteger,
    indexOffset As UInteger,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
```

```
    userData As Object,
    type As Type
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type) [▶ 508]

Reference

AdsConnection Class [▶ 148]

AddDeviceNotificationEx Overload [▶ 188]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.AddDeviceNotificationEx Method (UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotificationEx(
    uint indexGroup,
    uint indexOffset,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData,
    Type type,
    int[] args
)
```

VB

```
Public Function AddDeviceNotificationEx (
    indexGroup As UInteger,
    indexOffset As UInteger,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object,
    type As Type,
    args As Integer()
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 509]

Reference

AdsConnection Class [▶ 148]

AddDeviceNotificationEx Overload [▶ 188]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.3 AdsConnection.Close Method

Closes the AdsConnection [▶ 148]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Close()
```

VB

```
Public Sub Close
```

Implements

IConnection.Close. [▶ 50]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.4 AdsConnection.Connect Method

(Re)Connects the IConnection [▶ 46] when disconnected.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Connect()
```

VB

```
Public Function Connect As Boolean
```

Return Value

Type: Boolean
true if the AdsConnection [▶ 148] is reconnected, false otherwise.

Implements

IConnection.Connect. [▶ 50]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.5 AdsConnection.CreateSymbolLoader Method

Creates a new instance of the Symbol loader [▶ 979] with the specified mode.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IAdsSymbolLoader CreateSymbolLoader(
    ISession session,
    SymbolLoaderSettings settings
)
```

VB

```
Public Function CreateSymbolLoader (
    session As ISession,
    settings As SymbolLoaderSettings
) As IAdsSymbolLoader
```

Parameters

session	Type: TwinCAT.ISession [▶ 60] The session (for session orientated loades / symbols). Can be NULL if not present.
settings	Type: TwinCAT.Ads.SymbolLoaderSettings [▶ 615] The settings.

Return Value

Type: IAdsSymbolLoader [▶ 979]
The IAdsSymbolLoader [▶ 979] interface of the Symbol loader.

Exceptions

Exception	Condition
ObjectDisposedException	
ObjectDisposedException	

Remarks

The Symbol Loader (V2) supports the following modes [▶ 108]. Flat [▶ 108]The flat mode organizes the Symbols in a flat list. At the beginning this List caches only the root symbol objects, which can be enumerated. To access the sub elements like structure fields or array elements use the SubSymbols [▶ 1640] collection. The property get accessor generates the subsymbols lazy on the fly (performance optimized) and stores them internally as weak reference (memory optimized). This mode is available in all .NET

versions.VirtualTree [▶ 108]On top of the behaviour of the Flat [▶ 108], the virtual tree mode organizes the Symbols hierarchically with parent-child relationships. That eases the access to the hierarchical structure but needs slightly more preprocessing of the data. This mode is available in all .NET Versions. DynamicTree [▶ 108]The Dynamic tree mode organizes the Symbols hierarchically and (dynamically) creates struct members, array elements and enum fields on the fly. 'Dynamically' means here not only lazy creation like in Flat [▶ 108], but furthermore real creation of type safe .NET complex types/instances as represetantives of the TwinCAT Symbol objects/types. This feature is only available on platforms that support the Dynamic Language Runtime (DLR); actually all .NET Framework Version larger than 4.0. Virtual instances means, that all Symbols are ordered within a tree structure. For that symbol nodes that are not located on a fixed address, a Virtual Symbol will be created. Setting the virtualInstance parameter to 'false' means, that the located symbols will be returned in a flattened list.

Examples

The following sample shows how to create a dynamic version of the SymbolLoader V2. The dynamic symbol loader makes use of the Dynamic Language Runtime (DLR) of the .NET Framework. That means Structures, Arrays and Enumeration types and instances are generated 'on-the-fly' during symbol Browsing. These created dynamic objects are a one to one representation of the Symbol Server target objects (e.g the IEC61131 types on the PLC). Dynamic language features are only available from .NET4 upwards.

Dynamic Tree Mode

```
namespace Sample
{
    using System;
    using System.Diagnostics;
    using System.Threading;
    using TwinCAT;
    using TwinCAT.Ads;
    using TwinCAT.Ads.TypeSystem;
    using TwinCAT.Ads.ValueAccess;
    using TwinCAT.TypeSystem;
    using TwinCAT.TypeSystem.Generic;

    class SymbolBrowserProgramV2DynamicTree
    {

    #region CODE_SAMPLE_SIMPLEDYNAMIC
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        // Get the AdsAddress from command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        using (TcAdsClient client = new TcAdsClient())
        {
        // Connect to the target device
        client.Connect(address);

        // Usage of "dynamic" Type and Symbols (>= .NET4 only)
        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.DynamicTree);
        IAdsSymbolLoader dynLoader = (IAdsSymbolLoader)SymbolLoaderFactory.Create(client, settings);

        #endregion

        // Set the Default setting for Notifications
        dynLoader.DefaultNotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 200,
 2000);

        // Get the Symbols (Dynamic Symbols)
        dynamic dynamicSymbols = ((IDynamicSymbolLoader)dynLoader).SymbolsDynamic;

        dynamic adsPort = dynamicSymbols.TwinCAT_SystemInfoVarList._AppInfo.AdsPort;

        #region CODE_SAMPLE_SIMPLEDYNAMIC

        // Access Main Symbol with Dynamic Language Runtime support (DLR)
        // Dynamically created property "Main"
        //dynamic symMain = dynamicSymbols.Main;
```

```
        // Main is an 'VirtualSymbol' / Organizational unit that doesn't have a value
        // Calling ReadValue is not allowed
        //bool test = symMain.HasValue;
        //dynamic invalid = symMain.ReadValue();

        //Reading TaskInfo Value
        //With calling ReadValue() a 'snapshot' of the Symbols Instance is taken
        dynamic vTaskInfoArray = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo.ReadValue();

        // Getting the Snapshot time in UTC format
        DateTime timeStamp1 = vTaskInfoArray.UtcTimeStamp;

        // Getting TaskInfo Symbol for Task 1
        dynamic symTaskInfo1 = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo[1];

        // Getting CycleCount Symbol
        dynamic symCycleCount = symTaskInfo1.CycleCount;


        // Take Snapshot value of the ApplicationInfo struct
        dynamic vAppInfo = dynamicSymbols.TwinCAT_SystemInfoVarList._AppInfo.ReadValue();
        // Get the UTC Timestamp of the snapshot

        DateTime timeStamp2 = vAppInfo.UtcTimeStamp;

        // Access the ProjectName of the ApplicationInfo Snapshot (type-safe!)
        string projectNameValue = vAppInfo.ProjectName;

        // Reading the CycleCount Value
        uint cycleCountValue = symTaskInfo1.CycleCount.ReadValue();     // Taking a Value Snapshot
        #endregion

        // Registering for dynamic "ValueChanged" events for the Values
        // Using Default Notification settings
        symCycleCount.ValueChanged += new EventHandler<ValueChangedArgs>(cycleCount_ValueChanged);

        // Override default notification settings
        symTaskInfo1.NotificationSettings = new NotificationSettings(AdsTransMode.Cyclic,500,0);

        // Register for ValueChanged event.
        symTaskInfo1.ValueChanged += new EventHandler<ValueChangedArgs>(taskInfo1Value_ValueChanged)
; // Struct Type

        Thread.Sleep(10000); // Sleep main thread for 10 Seconds
        }
        Console.WriteLine("CycleCount Changed events received: {0}",_cycleCountEvents);
        Console.WriteLine("taskInfo1 Changed events received: {0}", _taskInfo1Events);

        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
    }

    static object _notificationSynchronizer = new object();
    static int _cycleCountEvents = 0;

    /// <summary>
    /// Handler function for the CycleCount ValueChanged event.
    /// </summary>
    /// <param name="sender">The sender.</param>
    /// <param name="e">The e.</param>
    static void cycleCount_ValueChanged(object sender, ValueChangedArgs e)
    {
        lock(_notificationSynchronizer)
        {
        Interlocked.Increment(ref _cycleCountEvents);
        // val is a type safe value of int!
        dynamic val = e.Value;
        uint intVal = val;

        DateTime changedTime = e.UtcRtime.ToLocalTime(); // Convert UTC to local time
        Console.WriteLine("CycleCount changed to: {0}, TimeStamp: {1}", intVal, changedTime.ToString
("HH:mm:ss:fff"));
        }
    }

    static int _taskInfo1Events = 0;

    /// <summary>
    /// Handler function for the TaskInfo ValueChanged event.
```

```
    /// </summary>
    /// <param name="sender">The sender.</param>
    /// <param name="e">The e.</param>
    static void taskInfo1Value_ValueChanged(object sender, ValueChangedArgs e)
    {
        lock (_notificationSynchronizer)
        {
        Interlocked.Increment(ref _taskInfo1Events);
        dynamic val = e.Value;
        DateTime changedTime = e.UtcRtime.ToLocalTime(); // Convert to local time

        // Val is a during Runtime created struct type and contains
        // the same Properties as related PLC object.
        int cycleTime = val.CycleTime;
        Console.WriteLine("TaskInfo1Value changed TimeStamp: {0}", changedTime.ToString("HH:mm:ss:ff
f"));
        }
    }
    }
}
```

The following sample shows how to create a static (non dynamic) version of the SymbolLoader V2. The static symbol loader in version 2 is a nearly code compatible version of the Dynamic Loader, only the dynamic creation of objects is not available. The reason for supporting this mode is that .NET Framework Versions lower than Version 4.0 (CLR2) doesn't support the Dynamic Language Runtime (DLR). The SymbolLoader V2 static object is supported from .NET 2.0 on.

Virtual Tree Mode

```
using System;
using System.Threading;
using System.Diagnostics;
using TwinCAT;
using TwinCAT.Ads;
using TwinCAT.TypeSystem;
using TwinCAT.TypeSystem.Generic;
using TwinCAT.Ads.ValueAccess;
using TwinCAT.Ads.TypeSystem;

namespace Sample
{
    class SymbolBrowserProgramV2VirtualTree
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        ConsoleLogger logger = new ConsoleLogger();

        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for start:");
        Console.ReadLine();

        //logger.Active = false;

        Stopwatch stopper = new Stopwatch();

        // Parse the command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        stopper.Start();

        using (TcAdsClient client = new TcAdsClient())
        {
        client.Synchronize = false;

        // Connect the AdsClient to the device target.
        client.Connect(address);

        // Creates the Symbol Objects as hierarchical tree
        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.VirtualTree, ValueA
ccessMode.IndexGroupOffsetPreferred);
        ISymbolLoader symbolLoader = SymbolLoaderFactory.Create(client, settings);

        // Dump Datatypes from Target Device
        Console.WriteLine(string.Format("Dumping '{0}' DataTypes:", symbolLoader.DataTypes.Count));
```

```
        foreach (IDataType type in symbolLoader.DataTypes)
        {
            logger.DumpType(type);
        }
        Console.WriteLine("");

        // Dump Symbols from target device
        Console.WriteLine("Dumping '{0}' Symbols:", symbolLoader.Symbols.Count);
        foreach (ISymbol symbol in symbolLoader.Symbols)
        {
            logger.DumpSymbol(symbol,0);
        }
        }
        stopper.Stop();
        TimeSpan elapsed = stopper.Elapsed;

        Console.WriteLine("");
        Console.WriteLine("Browsing complete tree: {0},
({1} DataTypes, {2} Symbols)",elapsed,logger.DataTypesCount,logger.DataTypesCount);
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
    }
```

Examples

The SymbolLoader V2 static object is supported from .NET 2.0 on.

Flat Mode

```
using System;
using System.Diagnostics;
using System.Threading;
using TwinCAT;
using TwinCAT.Ads;
using TwinCAT.Ads.TypeSystem;
using TwinCAT.Ads.ValueAccess;
using TwinCAT.TypeSystem;
using TwinCAT.TypeSystem.Generic;

namespace Sample
{
    class SymbolBrowserProgramV2Flat
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        ConsoleLogger logger = new ConsoleLogger();

        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for start:");
        Console.ReadLine();

        //logger.Active = false;

        Stopwatch stopper = new Stopwatch();

        // Parse the command line arguments
        AmsAddress address = ArgParser.Parse(args);

        stopper.Start();

        // Create the ADS Client
        using (TcAdsClient client = new TcAdsClient())
        {
        client.Synchronize = false;

        // Connect to Address
        client.Timeout = 30000;
        client.Connect(address);

        // Creates the Symbol Objects in Flat Mode (Flat list)
        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.Flat, ValueAccessMo
de.IndexGroupOffsetPreferred);
        ISymbolLoader symbolLoader = SymbolLoaderFactory.Create(client, settings);
```

```
        // Dump Datatypes from Target Device
        Console.WriteLine(string.Format("Dumping '{0}' DataTypes:",symbolLoader.DataTypes.Count));
        foreach (IDataType type in symbolLoader.DataTypes)
        {
            logger.DumpType(type);
        }

        Console.WriteLine("");

        // Dump Symbols from target device
        Console.WriteLine("Dumping '{0}' Symbols:",symbolLoader.Symbols.Count);
        foreach (ISymbol symbol in symbolLoader.Symbols)
        {
            logger.DumpSymbol(symbol,0);
        }
        }
        stopper.Stop();
        TimeSpan elapsed = stopper.Elapsed;

        Console.WriteLine("");
        Console.WriteLine("Browsing complete tree: {0},
({1} DataTypes, {2} Symbols)", elapsed, logger.DataTypesCount, logger.DataTypesCount);
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
    }
```

Examples

Argument Parser

```
public static class ArgParser
{
    /// <summary>
    /// Parses the arguments.
    /// </summary>
    /// <param name="args">The arguments.</param>
    /// <returns>AmsAddress.</returns>
    public static AmsAddress Parse(string[] args)
    {
    AmsNetId netId = AmsNetId.Local;
    int port = 851;

    if (args != null)
    {
        if (args.Length > 0 && args[0] != null)
        netId = AmsNetId.Parse(args[0]);

        if (args.Length > 1 && args[1] != null)
        port = int.Parse(args[1]);
    }
    return new AmsAddress(netId, port);
    }
}
```

Dumping Symbols

```
/// <summary>
/// Console logger
/// </summary>
public class ConsoleLogger
{
    public ConsoleLogger()
    {
    }
    bool _active = true;

    /// <summary>
    /// Gets or sets a value indicating whether this ConsoleLogger is active.
    /// </summary>
    /// <value><c>true</c> if active; otherwise, <c>false</c>.</value>
    public bool Active
    {
    get { return _active; }
    set
    {
        _active = value;
    }
    }
```

```
    int _dataTypes = 0;

    /// <summary>
    /// Gets the number of dumped dataTypes.
    /// </summary>
    /// <value>The data types count.</value>
    public int DataTypesCount
    {
    get { return _dataTypes; }
    }

    int _symbols = 0;

    /// <summary>
    /// Gets the number of dumped symbols
    /// </summary>
    /// <value>The symbols count.</value>
    public int SymbolsCount
    {
    get { return _symbols; }
    }


    /// <summary>
    /// Dumps the data type.
    /// </summary>
    /// <param name="dataType">Data Type.</param>
    public void DumpType(IDataType dataType)
    {
    WriteLine(string.Format("DataType: {0}, Category: {1}, Size: {2}", dataType.Name, dataType.Categ
ory, dataType.Size));

    switch (dataType.Category)
    {
        case DataTypeCategory.Alias:
        IAliasType alias = (IAliasType)dataType;
        WriteLine(GetPrefix(1) + string.Format("Alias BaseType: {0}", alias.BaseTypeName));
        break;

        case DataTypeCategory.Enum:

        //IEnumType<ushort> enumType = (IEnumType<ushort>)dataType;
        IEnumType enumType = (IEnumType)dataType;
        WriteLine(GetPrefix(1) + string.Format("Enum BaseType: {0}", enumType.BaseTypeName));

        foreach (IEnumValue enumValue in enumType.EnumValues)
        {
            WriteLine(GetPrefix(2) + string.Format("Name: {0}, Value: {1}", enumValue.Name, enumValu
e.Primitive));
        }
        break;
        case DataTypeCategory.Array:

        IArrayType arrayType = (IArrayType)dataType;
        int i = 0;

        foreach (IDimension dim in arrayType.Dimensions)
        {
            WriteLine(GetPrefix(2) + string.Format("{0}: LowerBound: {1}, Elements: {2}", i+
+, dim.LowerBound, dim.ElementCount));
        }
        break;
        case DataTypeCategory.Struct:
        IStructType structType = (IStructType)dataType;

        foreach (IMember member in structType.Members)
        {
            WriteLine(GetPrefix(2) + string.Format("Offset {0}: Name: {1}, Type: {2}", member.Offset
, member.InstanceName, member.TypeName));
        }
        break;
        default:
        break;
    }

    foreach (ITypeAttribute attribute in dataType.Attributes)
    {
        WriteLine(GetPrefix(1) + string.Format("{{ {0} : {1} }}", attribute.Name, attribute.Value));
    }
```

```
    if (!string.IsNullOrEmpty(dataType.Comment))
    {
        WriteLine(GetPrefix(1) + string.Format("Comment: {0}", dataType.Comment));
    }

    IRpcCallableType rpcCallable = dataType as IRpcCallableType;

    if (rpcCallable != null)
    {
        foreach (IRpcMethod rpcMethod in rpcCallable.RpcMethods)
        {
        if (string.IsNullOrEmpty(rpcMethod.Comment))
            WriteLine(GetPrefix(1) + string.Format("Method: {0}", rpcMethod));
        else
            WriteLine(GetPrefix(1) + string.Format("Method: {0}, Comment: {1}", rpcMethod, rpcMethod
.Comment));
        }
    }
    _dataTypes++;
    }

    /// <summary>
    /// Dumps the Datatype to Console
    /// </summary>
    /// <param name="dataType">DataType.</param>
    public void DumpType(ITcAdsDataType dataType)
    {
    // Dump the Attributes (PLC Metadata)
    foreach (ITypeAttribute attribute in dataType.Attributes)
    {
        WriteLine(GetPrefix(1) + string.Format("{{ {0} : {1} }}", attribute.Name, attribute.Value));
    }

    WriteLine(string.Format("DataType: {0}, Category: {1}, Size: {2}", dataType.Name, dataType.Categ
ory, dataType.Size));

    if (dataType.BaseType != null)
    {
        WriteLine(GetPrefix(1) + string.Format("BaseType: {0}", dataType.BaseType));
    }

    switch (dataType.Category)
    {
        case DataTypeCategory.Enum:
        foreach (IEnumValue enumValue in dataType.EnumValues)
        {
            WriteLine(GetPrefix(2) + string.Format("Name: {0}, Value: {1}", enumValue.Name, enumValu
e.Primitive));
        }
        break;
        case DataTypeCategory.Array:
        int i = 0;
        foreach (IDimension dim in dataType.Dimensions)
        {
            WriteLine(GetPrefix(2) + string.Format("{0}: LowerBound: {1}, Elements: {2}", i+
+, dim.LowerBound, dim.ElementCount));
        }
        break;
        case DataTypeCategory.Struct:
        foreach (ITcAdsSubItem subItem in dataType.SubItems)
        {
            WriteLine(GetPrefix(2) + string.Format("Offset {0}: Name: {1}, Type: {2}", subItem.Offse
t, subItem.SubItemName, subItem.Name));
        }
        break;
        default:
        break;
    }
    _dataTypes++;
    }

    /// <summary>
    /// Dump Symbol
    /// </summary>
    /// <param name="symbol">The symbol.</param>
    /// <param name="level">Output indentation level</param>
    public void DumpSymbol(ISymbol symbol, int level)
    {
    IDataType type = symbol.DataType as IDataType;
```

```
    foreach (ITypeAttribute attribute in symbol.Attributes)
    {
        WriteLine(GetPrefix(level) + string.Format("{{ {0} : {1} }}", attribute.Name, attribute.Valu
e));
    }

       WriteLine(GetPrefix(level) + string.Format("{0} : {1} (IG: 0x{2} IO: 0x{3} size:
{4})", symbol.InstanceName, symbol.TypeName, ((IAdsSymbol)symbol).IndexGroup.ToString("x"), ((IAdsSy
mbol)symbol).IndexOffset.ToString("x"), symbol.Size));

    if (symbol.Category == DataTypeCategory.Array)
    {
        IArrayInstance arrInstance = (IArrayInstance)symbol;
        IArrayType arrType = (IArrayType)symbol.DataType;

        int count = 0;
        level++;

        foreach (ISymbol arrayElement in arrInstance.Elements)
        {
        DumpSymbol(arrayElement, level);
        count++;

        if (count > 20) // Write only the first 20 to limit output
            break;
        }
    }
    else if (symbol.Category == DataTypeCategory.Struct)
    {
        IStructInstance structInstance = (IStructInstance)symbol;
        IStructType structType = (IStructType)symbol.DataType;

        level++;

        foreach (ISymbol member in structInstance.MemberInstances)
        {
        DumpSymbol(member, level);
        }
    }
    _symbols++;
    }

    /// <summary>
    /// Dumps the specified Symbol to the Console
    /// </summary>
    /// <param name="symbol">The symbol.</param>
    /// <param name="level">The level.</param>
    public void DumpSymbol(ITcAdsSymbol5 symbol, int level)
    {
    // Dump Attributes of the Symbol
    foreach (ITypeAttribute attribute in symbol.Attributes)
    {
        WriteLine(GetPrefix(level) + string.Format("{{ {0} : {1} }}", attribute.Name, attribute.Valu
e));
    }

    ITcAdsSymbolBrowser subSymbolProvider = (ITcAdsSymbolBrowser)symbol;

    // Dump The Symbol
    WriteLine(GetPrefix(level) + string.Format("{0} : {1} ({2}, IG: 0x{3} IO: 0x{4} size:
{6} subCount:
{5})", symbol.Name, symbol.TypeName, symbol.DataTypeId, symbol.IndexGroup.ToString("x"), symbol.Inde
xOffset.ToString("x"), subSymbolProvider.SubSymbols.Count, symbol.Size));
    level++;

    // Dump all SubSymbols with indentation
    foreach (ITcAdsSymbol5 subSymbol in ((ITcAdsSymbolBrowser)symbol).SubSymbols)
    {
        DumpSymbol(subSymbol, level);
    }
    _symbols++;
    }

    /// <summary>
    /// Dump namespace.
    /// </summary>
    /// <param name="ns">The namespace.</param>
    public void DumpNamespace(INamespace<IDataType> ns)
    {
    WriteLine("Namespace: {0}, DataTypes: {1}", ns.Name, ns.DataTypes.Count);
```

```
    foreach (IDataType type in ns.DataTypes)
    {
        DumpType(type);
    }
    }

    /// <summary>
    /// Get the indentation prefix
    /// </summary>
    /// <param name="level">The level.</param>
    /// <returns>System.String.</returns>
    public string GetPrefix(int level)
    {
    return "".PadLeft(level * 3);
    }

    /// <summary>
    /// Writes a line to the Console
    /// </summary>
    /// <param name="message">The message.</param>
    public void WriteLine(string message)
    {
    if (Active)
    {
        Console.WriteLine(message);
    }
    }

    /// <summary>
    /// Writes a line to the console
    /// </summary>
    /// <param name="format">The format.</param>
    /// <param name="args">The arguments.</param>
    public void WriteLine(string format, params object[] args)
    {
    if (Active)
    {
        Console.WriteLine(format,args);
    }
    }
}
```

Examples

The following sample shows how to call (Remote Procedures / Methods) with Virtual Symbols

RPC Call in Virtual Mode

```
class RpcCallVirtualProgram
{
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
    // Get the AdsAddress from command-line arguments
    AmsAddress address = ArgParser.Parse(args);

    using (TcAdsClient client = new TcAdsClient())
    {
        client.Synchronize = false;

        // Connect to the target device
        client.Connect(address);

        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.VirtualTree);
        ISymbolLoader loader = SymbolLoaderFactory.Create(client, settings);

        // Get the Symbols (Dynamic Symbols)

        IRpcStructInstance main = (IRpcStructInstance)loader.Symbols["MAIN"]; // Gets the MAIN Insta
nce of the PLC Program

        // Call a Method that has the following signature (within MAIN Program)
        /*  {attribute 'TcRpcEnable'}
        METHOD PUBLIC M_Add : INT
```

```
        VAR_INPUT
            i1 : INT := 0;
            i2 : INT := 0;
        END_VAR
        */

        short result = (short)main.InvokeRpcMethod("M_Add", new object[] {(short) 3, (short) 4});

        // Call a Method that has no parameter and returns VOID
        main.InvokeRpcMethod("M_Method1", new object[] {});

        //Browsing RpcMethods
        foreach(IRpcMethod method in main.RpcMethods)
        {
        string methodName = method.Name;

        foreach(IRpcMethodParameter parameter in method.Parameters)
        {
            string parameterName = parameter.Name;
            string parameterType = parameter.TypeName;
        }
        }
    }
    }
}
```

Examples

The following sample shows how to call (Remote Procedures / Methods) with Dynamic Symbols.

RPC Call in Dynamic Mode

```
namespace Sample
{
    using System;
    using System.Diagnostics;
    using System.Threading;
    using TwinCAT;
    using TwinCAT.Ads;
    using TwinCAT.Ads.TypeSystem;
    using TwinCAT.Ads.ValueAccess;
    using TwinCAT.TypeSystem;
    using TwinCAT.TypeSystem.Generic;

    class RpcCallDynamicProgram
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        // Get the AdsAddress from command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        using (TcAdsClient client = new TcAdsClient())
        {
        client.Synchronize = false;

        // Connect to the target device
        client.Connect(address);

        // Usage of "dynamic" Type and Symbols (>= .NET4 only)
        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.DynamicTree);
        ISymbolLoader dynLoader = SymbolLoaderFactory.Create(client, settings);

        // Get the Symbols (Dynamic Symbols)
        dynamic symbols = ((IDynamicSymbolLoader)dynLoader).SymbolsDynamic;
        dynamic main = symbols.Main; // Gets the MAIN Instance of the PLC Program

         // Call a Method that has the following signature (within MAIN Program)

        /*  {attribute 'TcRpcEnable'}
            METHOD PUBLIC M_Add : INT
            VAR_INPUT
            i1 : INT := 0;
            i2 : INT := 0;
            END_VAR
```

```
        */

        short result = main.M_Add(3,4);

        // Call a Method that has no parameter and returns VOID
        main.M_Method1();

        //Browsing Rpc Methods
        foreach (IRpcMethod method in main.RpcMethods)
        {
            string methodName = method.Name;

            foreach (IRpcMethodParameter parameter in method.Parameters)
            {
            string parameterName = parameter.Name;
            string parameterType = parameter.TypeName;
            }
        }
        }
    }
    }
}
```

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

TwinCAT.Ads.TypeSystem.SymbolLoaderFactory [▶ 1100]

5.2.5.2.6 AdsConnection.CreateVariableHandle Method

Generates a unique handle for an ADS variable.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int CreateVariableHandle(
    string variableName
)
```

VB

```
Public Function CreateVariableHandle (
    variableName As String
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable

Return Value

Type: Int32
The handle of the ADS Variable.

Implements

IAdsHandleAccess.CreateVariableHandle(String) [▶ 486]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.7 AdsConnection.DeleteDeviceNotification Method

Deletes an existing notification.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void DeleteDeviceNotification(
    int notificationHandle
)
```

VB

```
Public Sub DeleteDeviceNotification (
    notificationHandle As Integer
)
```

Parameters

notificationHandle	Type: System.Int32 Handle of the notification.

Implements

IAdsNotifications.DeleteDeviceNotification(Int32) [▶ 510]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.8 AdsConnection.DeleteVariableHandle Method

Releases the handle of a ADS variable again.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void DeleteVariableHandle(
    int variableHandle
)
```

VB

```
Public Sub DeleteVariableHandle (
    variableHandle As Integer
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable

Implements

IAdsHandleAccess.DeleteVariableHandle(Int32) [▶ 487]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.9 AdsConnection.Disconnect Method

Disconnects this IConnection [▶ 46].

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Disconnect()
```

VB

```
Public Function Disconnect As Boolean
```

Return Value

Type: Boolean
true if XXXX, false otherwise.

Implements

IConnection.Disconnect. [▶ 51]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.10 AdsConnection.Dispose Method

Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Dispose()
```

VB

```
Public Sub Dispose
```

Implements

IDisposable.Dispose.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.11 AdsConnection.InvokeRpcMethod Method

Overload List

	Name	Description
	InvokeRpcMethod(String, Int32, .Object.) [▶ 210]	Invokes the specified RPC Method
	InvokeRpcMethod(String, String, .Object.) [▶ 210]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 211]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 212]	Invokes the specified RPC Method.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.InvokeRpcMethod Method (String, Int32, .Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object InvokeRpcMethod(
    string symbolPath,
    int methodId,
    Object[] parameters
)
```

VB

```
Public Function InvokeRpcMethod (
    symbolPath As String,
    methodId As Integer,
    parameters As Object()
) As Object
```

Parameters

symbolPath	Type: System.String The symbol path.
methodId	Type: System.Int32 The method identifier.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
The return value of the Method (as object).

Implements

ITcAdsRpcInvoke.InvokeRpcMethod(String, Int32, .Object.) [▶ 537]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

InvokeRpcMethod Overload [▶ 209]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.InvokeRpcMethod Method (String, String, .Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object InvokeRpcMethod(
    string symbolPath,
    string methodName,
    Object[] parameters
)
```

VB

```
Public Function InvokeRpcMethod (
    symbolPath As String,
    methodName As String,
    parameters As Object()
) As Object
```

Parameters

symbolPath	Type: System.String The symbol path.
methodName	Type: System.String The method name.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
The return value of the Method (as object).

Implements

ITcAdsRpcInvoke.InvokeRpcMethod(String, String, .Object.) [▶ 538]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

InvokeRpcMethod Overload [▶ 209]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.InvokeRpcMethod Method (ITcAdsSymbol, Int32, .Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object InvokeRpcMethod(
    ITcAdsSymbol symbol,
    int methodId,
    Object[] parameters
)
```

VB

```
Public Function InvokeRpcMethod (
    symbol As ITcAdsSymbol,
    methodId As Integer,
    parameters As Object()
) As Object
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
methodId	Type: System.Int32 The method identifier / Virtual Function table index.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
The return value of the Method (as object).

Implements

ITcAdsRpcInvoke.InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 539]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

InvokeRpcMethod Overload [▶ 209]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.InvokeRpcMethod Method (ITcAdsSymbol, String, .Object.)

Invokes the specified RPC Method.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object InvokeRpcMethod(
    ITcAdsSymbol symbol,
    string methodName,
    Object[] parameters
)
```

VB

```
Public Function InvokeRpcMethod (
    symbol As ITcAdsSymbol,
    methodName As String,
    parameters As Object()
) As Object
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
methodName	Type: System.String Name of the method.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
The return value of the Method (as object).

Implements

ITcAdsRpcInvoke.InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 540]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

InvokeRpcMethod Overload [▶ 209]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.12 AdsConnection.Read Method

Overload List

	Name	Description
	Read(Int32, AdsStream) [▶ 214]	Reads data synchronously from an ADS device and writes it to the given stream.

	Name	Description
	Read(UInt32, UInt32, AdsStream) [▶ 215]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(Int32, AdsStream, Int32, Int32) [▶ 216]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, .Byte., Int32, Int32) [▶ 216]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 217]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, .Byte., Int32, Int32, Int32) [▶ 218]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, AdsStream, Int32, Int32, Int32) [▶ 219]	Reads data synchronously from an ADS device and writes it to the given stream.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Read Method (Int32, AdsStream)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Read(
    int variableHandle,
    AdsStream dataStream
)
```

VB

```
Public Function Read (
    variableHandle As Integer,
    dataStream As AdsStream
) As Integer
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.

Return Value

Type: Int32
Number of successfully returned data bytes.

Implements

IAdsHandleAccess.Read(Int32, AdsStream) [▶ 488]

Reference

AdsConnection Class [▶ 148]

Read Overload [▶ 213]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Read Method (UInt32, UInt32, AdsStream)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Read(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream
)
```

VB

```
Public Function Read (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.

Return Value

Type: Int32
Number of successfully returned data bytes.

Reference

AdsConnection Class [▶ 148]

Read Overload [▶ 213]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Read Method (Int32, AdsStream, Int32, Int32)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Read(
    int variableHandle,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Function Read (
    variableHandle As Integer,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
) As Integer
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Implements

IAdsHandleAccess.Read(Int32, AdsStream, Int32, Int32) [▶ 489]

Reference

AdsConnection Class [▶ 148]

Read Overload [▶ 213]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Read Method (UInt32, UInt32, .Byte., Int32, Int32)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Read(
    uint indexGroup,
    uint indexOffset,
    byte[] readBuffer,
    int offset,
    int length
)
```

VB

```
Public Function Read (
    indexGroup As UInteger,
    indexOffset As UInteger,
    readBuffer As Byte(),
    offset As Integer,
    length As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
readBuffer	Type: .System.Byte. The read buffer.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

Read Overload [▶ 213]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Read Method (UInt32, UInt32, AdsStream, Int32, Int32)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Read(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Function Read (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Reference

AdsConnection Class [▶ 148]

Read Overload [▶ 213]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Read Method (UInt32, UInt32, .Byte., Int32, Int32, Int32)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Read(
    uint indexGroup,
    uint indexOffset,
    byte[] readBuffer,
    int offset,
    int length,
    int timeout
)
```

VB

```
Public Function Read (
    indexGroup As UInteger,
    indexOffset As UInteger,
    readBuffer As Byte(),
    offset As Integer,
    length As Integer,
    timeout As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
readBuffer	Type: .System.Byte. The read buffer.
offset	Type: System.Int32 The offset.
length	Type: System.Int32 The length.
timeout	Type: System.Int32 The timeout.

Return Value

Type: Int32
Number of successfully returned data bytes.

Reference

AdsConnection Class [▶ 148]

Read Overload [▶ 213]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Read Method (UInt32, UInt32, AdsStream, Int32, Int32, Int32)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Read(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length,
    int timeout
)
```

VB

```
Public Function Read (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    timeout As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
offset	Type: System.Int32 The offset.
length	Type: System.Int32 The length.
timeout	Type: System.Int32 The timeout.

Return Value

Type: Int32
Number of successfully returned data bytes.

Reference

AdsConnection Class [▶ 148]

Read Overload [▶ 213]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.13 AdsConnection.ReadAny Method

Overload List

	Name	Description
	ReadAny(Int32, Type) [▶ 221]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(Int32, Type, .Int32.) [▶ 222]	Reads any.

	Name	Description
	ReadAny(UInt32, UInt32, Type) [▶ 223]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 223]	Reads data synchronously from an ADS device and writes it to an object. If the Type of the object to be read is a string type, the first element of the parameter args specifies the number of characters of the string. If the Type of the object to be read is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	ReadAny(UInt32, UInt32, Type, .Int32., Int32) [▶ 224]	Reads data synchronously from an ADS device and writes it to an object. If the Type of the object to be read is a string type, the first element of the parameter args specifies the number of characters of the string. If the Type of the object to be read is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadAny Method (Int32, Type)

Reads data synchronously from an ADS device and writes it to an object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadAny(
    int variableHandle,
    Type type
)
```

VB

```
Public Function ReadAny (
    variableHandle As Integer,
    type As Type
) As Object
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable.
type	Type: System.Type Type of the object to be read.

Return Value

Type: Object
The object the read data is written to.

Implements

IAdsAnyAccess.ReadAny(Int32, Type) [▶ 461]

Reference

AdsConnection Class [▶ 148]

ReadAny Overload [▶ 220]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadAny Method (Int32, Type, .Int32.)

Reads any.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadAny(
    int variableHandle,
    Type type,
    int[] args
)
```

VB

```
Public Function ReadAny (
    variableHandle As Integer,
    type As Type,
    args As Integer()
) As Object
```

Parameters

variableHandle	Type: System.Int32 The variable handle.
type	Type: System.Type The type.
args	Type: .System.Int32. The arguments.

Return Value

Type: Object
System.Object.

Implements

IAdsAnyAccess.ReadAny(Int32, Type, .Int32.) [▶ 461]

Reference

AdsConnection Class [▶ 148]

ReadAny Overload [▶ 220]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadAny Method (UInt32, UInt32, Type)

Reads data synchronously from an ADS device and writes it to an object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadAny(
    uint indexGroup,
    uint indexOffset,
    Type type
)
```

VB

```
Public Function ReadAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    type As Type
) As Object
```

Parameters

indexGroup	Type: System.UInt32 Index group of the ADS variable.
indexOffset	Type: System.UInt32 Index offset of the ADS variable.
type	Type: System.Type Type of the object to be read.

Return Value

Type: Object
The object the read data is written to.

Implements

IAdsAnyAccess.ReadAny(UInt32, UInt32, Type) [▶ 462]

Reference

AdsConnection Class [▶ 148]

ReadAny Overload [▶ 220]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadAny Method (UInt32, UInt32, Type, .Int32.)

Reads data synchronously from an ADS device and writes it to an object. If the Type of the object to be read is a string type, the first element of the parameter args specifies the number of characters of the string. If the Type of the object to be read is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadAny(
    uint indexGroup,
    uint indexOffset,
    Type type,
    int[] args
)
```

VB

```
Public Function ReadAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    type As Type,
    args As Integer()
) As Object
```

Parameters

indexGroup	Type: System.UInt32 Index group of the ADS variable.
indexOffset	Type: System.UInt32 Index offset of the ADS variable.
type	Type: System.Type Type of the object to be read.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Object
The object the read data is written to.

Implements

IAdsAnyAccess.ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 463]

Reference

AdsConnection Class [▶ 148]

ReadAny Overload [▶ 220]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadAny Method (UInt32, UInt32, Type, .Int32., Int32)

Reads data synchronously from an ADS device and writes it to an object. If the Type of the object to be read is a string type, the first element of the parameter args specifies the number of characters of the string. If the Type of the object to be read is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadAny(
    uint indexGroup,
    uint indexOffset,
    Type type,
    int[] args,
    int timeout
)
```

VB

```
Public Function ReadAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    type As Type,
    args As Integer(),
    timeout As Integer
) As Object
```

Parameters

indexGroup	Type: System.UInt32 Index group of the ADS variable.
indexOffset	Type: System.UInt32 Index offset of the ADS variable.
type	Type: System.Type Type of the object to be read.
args	Type: .System.Int32. Additional arguments.
timeout	Type: System.Int32 The timeout.

Return Value

Type: Object
The object the read data is written to.

Reference

AdsConnection Class [▶ 148]

ReadAny Overload [▶ 220]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.14 AdsConnection.ReadAnyString Method

Overload List

	Name	Description
	ReadAnyString(Int32, Int32, Encoding) [▶ 226]	Reads the string.

	Name	Description
	ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 227]	Reads the string

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadAnyString Method (Int32, Int32, Encoding)

Reads the string.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ReadAnyString(
    int variableHandle,
    int len,
    Encoding encoding
)
```

VB

```
Public Function ReadAnyString (
    variableHandle As Integer,
    len As Integer,
    encoding As Encoding
) As String
```

Parameters

variableHandle	Type: System.Int32 The variable handle.
len	Type: System.Int32 The length.
encoding	Type: System.Text.Encoding The encoding.

Return Value

Type: String
System.String.

Implements

IAdsAnyAccess.ReadAnyString(Int32, Int32, Encoding) [▶ 464]

Exceptions

Exception	Condition
ObjectDisposedException	

Exception	Condition
ObjectDisposedException	

Remarks

ASCII Encoding expected

Reference

AdsConnection Class [▶ 148]

ReadAnyString Overload [▶ 225]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadAnyString Method (UInt32, UInt32, Int32, Encoding)

Reads the string

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ReadAnyString(
    uint indexGroup,
    uint indexOffset,
    int len,
    Encoding encoding
)
```

VB

```
Public Function ReadAnyString (
    indexGroup As UInteger,
    indexOffset As UInteger,
    len As Integer,
    encoding As Encoding
) As String
```

Parameters

indexGroup	Type: System.UInt32 The index group.
indexOffset	Type: System.UInt32 The index offset.
len	Type: System.Int32 The length.
encoding	Type: System.Text.Encoding The encoding.

Return Value

Type: String
System.String.

Implements

IAdsAnyAccess.ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 465]

Exceptions

Exception	Condition
ObjectDisposedException	
ObjectDisposedException	

Remarks

ASCII Encoding expected

Reference

AdsConnection Class [▶ 148]

ReadAnyString Overload [▶ 225]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.15 AdsConnection.ReadDeviceInfo Method

Reads the identification and version number of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DeviceInfo ReadDeviceInfo()
```

VB

```
Public Function ReadDeviceInfo As DeviceInfo
```

Return Value

Type: DeviceInfo [▶ 455]
DeviceInfo struct containing the name of the device and the version information.

Implements

IAdsConnection.ReadDeviceInfo. [▶ 483]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.16 AdsConnection.ReadState Method

Overload List

	Name	Description
	ReadState. [▶ 229]	Reads the ADS status and the device status from an ADS server.
	ReadState(Int32) [▶ 229]	Reads the ADS status and the device status from an ADS server.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadState Method

Reads the ADS status and the device status from an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public StateInfo ReadState()
```

VB

```
Public Function ReadState As StateInfo
```

Return Value

Type: StateInfo [▶ 598]
The ADS statue and device status.

Reference

AdsConnection Class [▶ 148]

ReadState Overload [▶ 229]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadState Method (Int32)

Reads the ADS status and the device status from an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public StateInfo ReadState(
    int timeout
)
```

VB

```
Public Function ReadState (
    timeout As Integer
) As StateInfo
```

Parameters

timeout	Type: System.Int32 The timeout.

Return Value

Type: StateInfo [▶ 598]
The ADS statue and device status.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

ReadState Overload [▶ 229]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.17 AdsConnection.ReadSymbol Method

Overload List

	Name	Description
	ReadSymbol(ITcAdsSymbol) [▶ 230]	Reads the value of a symbol and returns it as an object. Strings and all primitive datatypes(UInt32, Int32, Bool etc.) are supported. Arrays and structures cannot be read.
	ReadSymbol(String, Type, Boolean) [▶ 231]	Reads the value of a symbol and returns the value as object. The parameter type must have the same layout as the ADS symbol.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadSymbol Method (ITcAdsSymbol)

Reads the value of a symbol and returns it as an object. Strings and all primitive datatypes(UInt32, Int32, Bool etc.) are supported. Arrays and structures cannot be read.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadSymbol(
    ITcAdsSymbol symbol
)
```

VB

```
Public Function ReadSymbol (
    symbol As ITcAdsSymbol
) As Object
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol that should be read.

Return Value

Type: Object
The value of the symbol as an object.

Reference

AdsConnection Class [▶ 148]

ReadSymbol Overload [▶ 230]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadSymbol Method (String, Type, Boolean)

Reads the value of a symbol and returns the value as object. The parameter type must have the same layout as the ADS symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadSymbol(
    string name,
    Type type,
    bool reloadSymbolInfo
)
```

VB

```
Public Function ReadSymbol (
    name As String,
    type As Type,
    reloadSymbolInfo As Boolean
) As Object
```

Parameters

name	Type: System.String Name of the ADS symbol.
type	Type: System.Type Managed type of the ADS symbol.

reloadSymbolInfo	Type: System.Boolean If reload is true previously stored symbol information is cleared. As a consequence the symbol information must be obtained from the ADS server again.

Return Value

Type: Object
Value of the symbol

Reference

AdsConnection Class [▶ 148]

ReadSymbol Overload [▶ 230]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.18 AdsConnection.ReadSymbolInfo Method

Call this method to obtain information about the individual symbols (variables) in ADS devices.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ITcAdsSymbol ReadSymbolInfo(
    string name
)
```

VB

```
Public Function ReadSymbolInfo (
    name As String
) As ITcAdsSymbol
```

Parameters

name	Type: System.String Name of the symbol.

Return Value

Type: ITcAdsSymbol [▶ 549]
A ITcAdsSymbol containing the requested symbol information or null if symbol could not be found.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.19 AdsConnection.ReadWrite Method

Overload List

	Name	Description
	ReadWrite(UInt32, UInt32, AdsStream, AdsStream) [▶ 233]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, AdsStream, AdsStream, Int32) [▶ 234]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 235]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32) [▶ 236]	Writes data synchronously to an ADS device and then reads data from this device.
	ReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 237]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32) [▶ 239]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32) [▶ 240]	Writes data synchronously to an ADS device and then Reads data from this device.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadWrite Method (UInt32, UInt32, AdsStream, AdsStream)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReadWrite(
    uint indexGroup,
    uint indexOffset,
```

```
    AdsStream rdDataStream,
    AdsStream wrDataStream
)
```

VB

```
Public Function ReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    rdDataStream As AdsStream,
    wrDataStream As AdsStream
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.

Return Value

Type: Int32
Number of successfully returned data bytes.

Reference

AdsConnection Class [▶ 148]

ReadWrite Overload [▶ 233]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadWrite Method (UInt32, UInt32, AdsStream, AdsStream, Int32)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReadWrite(
    uint indexGroup,
    uint indexOffset,
    AdsStream rdDataStream,
    AdsStream wrDataStream,
    int timeout
)
```

VB

```
Public Function ReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    rdDataStream As AdsStream,
    wrDataStream As AdsStream,
    timeout As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
timeout	Type: System.Int32 The timeout.

Return Value

Type: Int32
Number of successfully returned data bytes.

Reference

AdsConnection Class [▶ 148]

ReadWrite Overload [▶ 233]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadWrite Method (Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReadWrite(
    int variableHandle,
    AdsStream rdDataStream,
    int rdOffset,
    int rdLength,
    AdsStream wrDataStream,
    int wrOffset,
    int wrLength
)
```

VB

```
Public Function ReadWrite (
    variableHandle As Integer,
    rdDataStream As AdsStream,
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As AdsStream,
    wrOffset As Integer,
    wrLength As Integer
) As Integer
```

Parameters

variableHandle	Type: System.Int32 Variable handle.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Implements

IAdsHandleAccess.ReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 490]

Reference

AdsConnection Class [▶ 148]

ReadWrite Overload [▶ 233]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadWrite Method (UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32)

Writes data synchronously to an ADS device and then reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReadWrite(
    uint indexGroup,
    uint indexOffset,
    byte[] readBuffer,
    int rdOffset,
    int rdLength,
    byte[] writeBuffer,
    int wrOffset,
    int wrLength
)
```

VB

```
Public Function ReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    readBuffer As Byte(),
    rdOffset As Integer,
    rdLength As Integer,
    writeBuffer As Byte(),
    wrOffset As Integer,
    wrLength As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
readBuffer	Type: .System.Byte. The read buffer.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
writeBuffer	Type: .System.Byte. The write buffer.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

ReadWrite Overload [▶ 233]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadWrite Method (UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReadWrite(
    uint indexGroup,
    uint indexOffset,
    AdsStream rdDataStream,
    int rdOffset,
    int rdLength,
    AdsStream wrDataStream,
    int wrOffset,
    int wrLength
)
```

VB

```
Public Function ReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    rdDataStream As AdsStream,
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As AdsStream,
    wrOffset As Integer,
    wrLength As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Reference

AdsConnection Class [▶ 148]

ReadWrite Overload [▶ 233]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadWrite Method (UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReadWrite(
    uint indexGroup,
    uint indexOffset,
    byte[] readBuffer,
    int rdOffset,
    int rdLength,
    byte[] writeBuffer,
    int wrOffset,
    int wrLength,
    int timeout
)
```

VB

```
Public Function ReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    readBuffer As Byte(),
    rdOffset As Integer,
    rdLength As Integer,
    writeBuffer As Byte(),
    wrOffset As Integer,
    wrLength As Integer,
    timeout As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
readBuffer	Type: .System.Byte. The read buffer.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
writeBuffer	Type: .System.Byte. The write buffer.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.
timeout	Type: System.Int32 The timeout.

Return Value

Type: Int32
Number of successfully returned data bytes.

Reference

AdsConnection Class [▶ 148]

ReadWrite Overload [▶ 233]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ReadWrite Method (UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReadWrite(
    uint indexGroup,
    uint indexOffset,
    AdsStream rdDataStream,
    int rdOffset,
    int rdLength,
    AdsStream wrDataStream,
    int wrOffset,
    int wrLength,
    int timeout
)
```

VB

```
Public Function ReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    rdDataStream As AdsStream,
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As AdsStream,
    wrOffset As Integer,
    wrLength As Integer,
    timeout As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.

wrLength	Type: System.Int32 Length of the data in wrDataStream.
timeout	Type: System.Int32 The timeout.

Return Value

Type: Int32
Number of successfully returned data bytes.

Reference

AdsConnection Class [▶ 148]

ReadWrite Overload [▶ 233]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.20 AdsConnection.TryAddDeviceNotification Method

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryAddDeviceNotification(
    string variableName,
    AdsStream dataStream,
    int offset,
    int length,
    NotificationSettings settings,
    Object userData,
    out uint handle
)
```

VB

```
Public Function TryAddDeviceNotification (
    variableName As String,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    settings As NotificationSettings,
    userData As Object,
    <OutAttribute> ByRef handle As UInteger
) As AdsErrorCode
```

Parameters

variableName	Type: System.String Name of the ADS variable.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

settings	Type: TwinCAT.Ads.NotificationSettings [▶ 573] The settings.
userData	Type: System.Object This object can be used to store user specific data.
handle	Type: System.UInt32. The handle.

Return Value

Type: AdsErrorCode [▶ 305]
The ADS ErrorCode.

Implements

IAdsNotifications.TryAddDeviceNotification(String, AdsStream, Int32, Int32, NotificationSettings, Object, UInt32.) [▶ 511]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.21 AdsConnection.TryAddDeviceNotificationEx Method

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryAddDeviceNotificationEx(
    string variableName,
    NotificationSettings settings,
    Object userData,
    Type type,
    int[] args,
    out uint handle
)
```

VB

```
Public Function TryAddDeviceNotificationEx (
    variableName As String,
    settings As NotificationSettings,
    userData As Object,
    type As Type,
    args As Integer(),
    <OutAttribute> ByRef handle As UInteger
) As AdsErrorCode
```

Parameters

variableName	Type: System.String Name of the ADS variable.
settings	Type: TwinCAT.Ads.NotificationSettings [▶ 573] The settings.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.
args	Type: .System.Int32. Additional arguments.
handle	Type: System.UInt32. The handle.

Return Value

Type: AdsErrorCode [▶ 305]
The handle of the notification.

Implements

IAdsNotifications.TryAddDeviceNotificationEx(String, NotificationSettings, Object, Type, .Int32., UInt32.) [▶ 512]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.22 AdsConnection.TryDeleteDeviceNotification Method

Deletes an existing notification.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryDeleteDeviceNotification(
    uint notificationHandle
)
```

VB

```
Public Function TryDeleteDeviceNotification (
    notificationHandle As UInteger
) As AdsErrorCode
```

Parameters

notificationHandle	Type: System.UInt32 Handle of the notification.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Implements

IAdsNotifications.TryDeleteDeviceNotification(UInt32) [▶ 513]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.23 AdsConnection.TryInvokeRpcMethod Method

Overload List

	Name	Description
	TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 245]	Invokes the specified RPC Method
	TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 246]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 247]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 248]	Invokes the specified RPC Method.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryInvokeRpcMethod Method (String, Int32, .Object., Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryInvokeRpcMethod(
    string symbolPath,
    int methodId,
    Object[] parameters,
    out Object retValue
)
```

VB

```
Public Function TryInvokeRpcMethod (
    symbolPath As String,
    methodId As Integer,
    parameters As Object(),
    <OutAttribute> ByRef retValue As Object
) As AdsErrorCode
```

Parameters

symbolPath	Type: System.String The symbol path.
methodId	Type: System.Int32 The method identifier.
parameters	Type: .System.Object. The parameters.
retValue	Type: System.Object. The return value of the RPC method as object.

Return Value

Type: AdsErrorCode [▶ 305]
The ADS Error Code.

Implements

ITcAdsRpcInvoke.TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 541]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryInvokeRpcMethod Overload [▶ 244]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryInvokeRpcMethod Method (String, String, .Object., Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryInvokeRpcMethod(
    string symbolPath,
    string methodName,
    Object[] parameters,
    out Object retValue
)
```

VB

```
Public Function TryInvokeRpcMethod (
    symbolPath As String,
    methodName As String,
    parameters As Object(),
    <OutAttribute> ByRef retValue As Object
) As AdsErrorCode
```

Parameters

symbolPath	Type: System.String The symbol path.
methodName	Type: System.String The method name.
parameters	Type: .System.Object. The parameters.
retValue	Type: System.Object. The return value of the RPC method as object.

Return Value

Type: AdsErrorCode [▶ 305]
The ADS Error Code.

Implements

ITcAdsRpcInvoke.TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 542]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryInvokeRpcMethod Overload [▶ 244]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryInvokeRpcMethod Method (ITcAdsSymbol, Int32, .Object., Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryInvokeRpcMethod(
    ITcAdsSymbol symbol,
    int methodId,
    Object[] parameters,
    out Object retValue
)
```

VB

```
Public Function TryInvokeRpcMethod (
    symbol As ITcAdsSymbol,
    methodId As Integer,
    parameters As Object(),
    <OutAttribute> ByRef retValue As Object
) As AdsErrorCode
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
methodId	Type: System.Int32 The method identifier / Virtual Function table index.
parameters	Type: .System.Object. The parameters.
retValue	Type: System.Object. The return value of the RPC method as object.

Return Value

Type: AdsErrorCode [▶ 305]
The ADS Error Code.

Implements

ITcAdsRpcInvoke.TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 543]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryInvokeRpcMethod Overload [▶ 244]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryInvokeRpcMethod Method (ITcAdsSymbol, String, .Object., Object.)

Invokes the specified RPC Method.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryInvokeRpcMethod(
    ITcAdsSymbol symbol,
    string methodName,
    Object[] parameters,
    out Object retValue
)
```

VB

```
Public Function TryInvokeRpcMethod (
    symbol As ITcAdsSymbol,
    methodName As String,
    parameters As Object(),
    <OutAttribute> ByRef retValue As Object
) As AdsErrorCode
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
methodName	Type: System.String Name of the method.
parameters	Type: .System.Object. The parameters.
retValue	Type: System.Object. The return value of the RPC method as object.

Return Value

Type: AdsErrorCode [▶ 305]
The ADS Error Code.

Implements

ITcAdsRpcInvoke.TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 544]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryInvokeRpcMethod Overload [▶ 244]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.24 AdsConnection.TryRead Method

Overload List

	Name	Description
	TryRead(Int32, AdsStream, Int32, Int32, Int32.) [▶ 249]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, .Byte., Int32, Int32, Int32.) [▶ 250]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, AdsStream, Int32, Int32, Int32.) [▶ 251]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, .Byte., Int32, Int32, Int32, Int32.) [▶ 252]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, AdsStream, Int32, Int32, Int32, Int32.) [▶ 253]	Reads data synchronously from an ADS device and writes it to the given stream.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryRead Method (Int32, AdsStream, Int32, Int32, Int32.)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryRead(
    int variableHandle,
    AdsStream dataStream,
    int offset,
    int length,
    out int readBytes
)
```

VB

```
Public Function TryRead (
    variableHandle As Integer,
    dataStream As AdsStream,
    offset As Integer,
```

```
    length As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
readBytes	Type: System.Int32. Number of successfully returned data bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode

Implements

IAdsHandleAccess.TryRead(Int32, AdsStream, Int32, Int32, Int32.) [▶ 491]

Reference

AdsConnection Class [▶ 148]

TryRead Overload [▶ 249]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryRead Method (UInt32, UInt32, .Byte., Int32, Int32, Int32.)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryRead(
    uint indexGroup,
    uint indexOffset,
    byte[] readBuffer,
    int offset,
    int length,
    out int readBytes
)
```

VB

```
Public Function TryRead (
    indexGroup As UInteger,
    indexOffset As UInteger,
    readBuffer As Byte(),
    offset As Integer,
```

```
    length As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
readBuffer	Type: .System.Byte. The read buffer.
offset	Type: System.Int32 The offset.
length	Type: System.Int32 The length.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ObjectDisposedException	
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryRead Overload [▶ 249]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryRead Method (UInt32, UInt32, AdsStream, Int32, Int32, Int32.)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryRead(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length,
    out int readBytes
)
```

VB

```
Public Function TryRead (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
offset	Type: System.Int32 The offset.
length	Type: System.Int32 The length.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryRead Overload [▶ 249]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryRead Method (UInt32, UInt32, .Byte., Int32, Int32, Int32, Int32.)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryRead(
    uint indexGroup,
    uint indexOffset,
    byte[] readBuffer,
    int offset,
    int length,
```

```
    int timeout,
    out int readBytes
)
```

VB

```
Public Function TryRead (
    indexGroup As UInteger,
    indexOffset As UInteger,
    readBuffer As Byte(),
    offset As Integer,
    length As Integer,
    timeout As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
readBuffer	Type: .System.Byte. The read buffer.
offset	Type: System.Int32 The offset.
length	Type: System.Int32 The length.
timeout	Type: System.Int32 The timeout.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

AdsConnection Class [▶ 148]

TryRead Overload [▶ 249]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryRead Method (UInt32, UInt32, AdsStream, Int32, Int32, Int32, Int32.)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryRead(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
```

```
    int offset,
    int length,
    int timeout,
    out int readBytes
)
```

VB

```
Public Function TryRead (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    timeout As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
offset	Type: System.Int32 The offset.
length	Type: System.Int32 The length.
timeout	Type: System.Int32 The timeout.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

AdsConnection Class [▶ 148]

TryRead Overload [▶ 249]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.25 AdsConnection.TryReadState Method

Overload List

	Name	Description
	TryReadState(StateInfo.) [▶ 255]	Reads the ADS status and the device status from an ADS server. Unlike the ReadState method this method does not call an exception on failure. Instead an AdsErrorCode is returned. If the return value is equal to AdsErrorCode.NoError the call was successful.

	Name	Description
	TryReadState(Int32, StateInfo.) [▶ 255]	Reads the ADS status and the device status from an ADS server. Unlike the ReadState method this method does not call an exception on failure. Instead an AdsErrorCode is returned. If the return value is equal to AdsErrorCode.NoError the call was successfull.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryReadState Method (StateInfo.)

Reads the ADS status and the device status from an ADS server. Unlike the ReadState method this method does not call an exception on failure. Instead an AdsErrorCode is returned. If the return value is equal to AdsErrorCode.NoError the call was successful.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReadState(
    out StateInfo stateInfo
)
```

VB

```
Public Function TryReadState (
    <OutAttribute> ByRef stateInfo As StateInfo
) As AdsErrorCode
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598]. The ADS statue and device status.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode of the ads read state call. Check for AdsErrorCode.NoError to see if call was successfull.

Reference

AdsConnection Class [▶ 148]

TryReadState Overload [▶ 254]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryReadState Method (Int32, StateInfo.)

Reads the ADS status and the device status from an ADS server. Unlike the ReadState method this method does not call an exception on failure. Instead an AdsErrorCode is returned. If the return value is equal to AdsErrorCode.NoError the call was successfull.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReadState(
    int timeout,
    out StateInfo stateInfo
)
```

VB

```
Public Function TryReadState (
    timeout As Integer,
    <OutAttribute> ByRef stateInfo As StateInfo
) As AdsErrorCode
```

Parameters

timeout	Type: System.Int32 The timeout.
stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598]. The ADS statue and device status.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode of the ads read state call. Check for AdsErrorCode.NoError to see if call was successfull.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryReadState Overload [▶ 254]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.26 AdsConnection.TryReadWrite Method

Overload List

	Name	Description
	TryReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 257]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, .Byte.,	Writes data synchronously to an ADS device and then Reads data from this device.

	Name	Description
	Int32, Int32, .Byte., Int32, Int32, Int32.) [▶ 258]	
	TryReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 260]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32, Int32.) [▶ 261]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32, Int32.) [▶ 262]	Writes data synchronously to an ADS device and then Reads data from this device.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryReadWrite Method (Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReadWrite(
    int variableHandle,
    AdsStream rdDataStream,
    int rdOffset,
    int rdLength,
    AdsStream wrDataStream,
    int wrOffset,
    int wrLength,
    out int readBytes
)
```

VB

```
Public Function TryReadWrite (
    variableHandle As Integer,
    rdDataStream As AdsStream,
    rdOffset As Integer,
```

```
    rdLength As Integer,
    wrDataStream As AdsStream,
    wrOffset As Integer,
    wrLength As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

variableHandle	Type: System.Int32 Variable handle.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Implements

IAdsHandleAccess.TryReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 492]

Reference

AdsConnection Class [▶ 148]

TryReadWrite Overload [▶ 256]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryReadWrite Method (UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32.)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReadWrite(
    uint indexGroup,
    uint indexOffset,
    byte[] readBuffer,
```

```
    int rdOffset,
    int rdLength,
    byte[] writeBuffer,
    int wrOffset,
    int wrLength,
    out int readBytes
)
```

VB

```
Public Function TryReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    readBuffer As Byte(),
    rdOffset As Integer,
    rdLength As Integer,
    writeBuffer As Byte(),
    wrOffset As Integer,
    wrLength As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
readBuffer	Type: .System.Byte. The read buffer.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
writeBuffer	Type: .System.Byte. The write buffer.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ObjectDisposedException	
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryReadWrite Overload [▶ 256]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryReadWrite Method (UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReadWrite(
    uint indexGroup,
    uint indexOffset,
    AdsStream rdDataStream,
    int rdOffset,
    int rdLength,
    AdsStream wrDataStream,
    int wrOffset,
    int wrLength,
    out int readBytes
)
```

VB

```
Public Function TryReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    rdDataStream As AdsStream,
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As AdsStream,
    wrOffset As Integer,
    wrLength As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryReadWrite Overload [▶ 256]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryReadWrite Method (UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32, Int32.)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReadWrite(
    uint indexGroup,
    uint indexOffset,
    byte[] readBuffer,
    int rdOffset,
    int rdLength,
    byte[] writeBuffer,
    int wrOffset,
    int wrLength,
    int timeout,
    out int readBytes
)
```

VB

```
Public Function TryReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    readBuffer As Byte(),
    rdOffset As Integer,
    rdLength As Integer,
    writeBuffer As Byte(),
    wrOffset As Integer,
    wrLength As Integer,
    timeout As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
readBuffer	Type: .System.Byte. The read buffer.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.

rdLength	Type: System.Int32 Length of the data in rdDataStream.
writeBuffer	Type: .System.Byte. The write buffer.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.
timeout	Type: System.Int32 The timeout.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

AdsConnection Class [▶ 148]

TryReadWrite Overload [▶ 256]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryReadWrite Method (UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32, Int32.)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReadWrite(
    uint indexGroup,
    uint indexOffset,
    AdsStream rdDataStream,
    int rdOffset,
    int rdLength,
    AdsStream wrDataStream,
    int wrOffset,
    int wrLength,
    int timeout,
    out int readBytes
)
```

VB

```
Public Function TryReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    rdDataStream As AdsStream,
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As AdsStream,
    wrOffset As Integer,
    wrLength As Integer,
```

```
    timeout As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.
timeout	Type: System.Int32 The timeout.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

AdsConnection Class [▶ 148]

TryReadWrite Overload [▶ 256]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.27 AdsConnection.TryWrite Method

Overload List

	Name	Description
	TryWrite(Int32, AdsStream, Int32, Int32) [▶ 264]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, .Byte., Int32, Int32) [▶ 265]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 266]	Writes data synchronously to an ADS device.

	Name	Description
	TryWrite(UInt32, UInt32, .Byte., Int32, Int32, Int32) [▶ 267]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, AdsStream, Int32, Int32, Int32) [▶ 268]	Writes data synchronously to an ADS device.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryWrite Method (Int32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWrite(
    int variableHandle,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Function TryWrite (
    variableHandle As Integer,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
) As AdsErrorCode
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Implements

IAdsHandleAccess.TryWrite(Int32, AdsStream, Int32, Int32) [▶ 493]

Reference

AdsConnection Class [▶ 148]

TryWrite Overload [▶ 263]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryWrite Method (UInt32, UInt32, .Byte., Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWrite(
    uint indexGroup,
    uint indexOffset,
    byte[] writeBuffer,
    int offset,
    int length
)
```

VB

```
Public Function TryWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    writeBuffer As Byte(),
    offset As Integer,
    length As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
writeBuffer	Type: .System.Byte. The write buffer.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ObjectDisposedException	
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryWrite Overload [▶ 263]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryWrite Method (UInt32, UInt32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWrite(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Function TryWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryWrite Overload [▶ 263]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryWrite Method (UInt32, UInt32, .Byte., Int32, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWrite(
    uint indexGroup,
    uint indexOffset,
    byte[] writeStream,
    int offset,
    int length,
    int timeout
)
```

VB

```
Public Function TryWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    writeStream As Byte(),
    offset As Integer,
    length As Integer,
    timeout As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
writeStream	Type: .System.Byte. The write stream.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
timeout	Type: System.Int32 The timeout.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

AdsConnection Class [▶ 148]

TryWrite Overload [▶ 263]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryWrite Method (UInt32, UInt32, AdsStream, Int32, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWrite(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length,
    int timeout
)
```

VB

```
Public Function TryWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    timeout As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
timeout	Type: System.Int32 The timeout.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

AdsConnection Class [▶ 148]

TryWrite Overload [▶ 263]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.28 AdsConnection.TryWriteControl Method

Overload List

	Name	Description
	TryWriteControl(StateInfo) [▶ 269]	Changes the ADS status and the device status of an ADS server.
	TryWriteControl(StateInfo, Int32) [▶ 270]	Changes the ADS status and the device status of an ADS server.
	TryWriteControl(StateInfo, AdsStream, Int32, Int32) [▶ 271]	Changes the ADS status and the device status of an ADS server.
	TryWriteControl(StateInfo, AdsStream, Int32, Int32, Int32) [▶ 272]	Changes the ADS status and the device status of an ADS server.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryWriteControl Method (StateInfo)

Changes the ADS status and the device status of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWriteControl(
    StateInfo stateInfo
)
```

VB

```
Public Function TryWriteControl (
    stateInfo As StateInfo
) As AdsErrorCode
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598] New ADS status and device status.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryWriteControl Overload [▶ 269]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryWriteControl Method (StateInfo, Int32)

Changes the ADS status and the device status of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWriteControl(
    StateInfo stateInfo,
    int timeout
)
```

VB

```
Public Function TryWriteControl (
    stateInfo As StateInfo,
    timeout As Integer
) As AdsErrorCode
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598] New ADS status and device status.
timeout	Type: System.Int32 The timeout.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryWriteControl Overload [▶ 269]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryWriteControl Method (StateInfo, AdsStream, Int32, Int32)

Changes the ADS status and the device status of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWriteControl(
    StateInfo stateInfo,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Function TryWriteControl (
    stateInfo As StateInfo,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
) As AdsErrorCode
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598] New ADS status and device status.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be sent to the ADS device
offset	Type: System.Int32 Offset of the data in the stream.
length	Type: System.Int32 Length of the data in the stream.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryWriteControl Overload [▶ 269]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.TryWriteControl Method (StateInfo, AdsStream, Int32, Int32, Int32)

Changes the ADS status and the device status of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWriteControl(
    StateInfo stateInfo,
    AdsStream dataStream,
    int offset,
    int length,
    int timeout
)
```

VB

```
Public Function TryWriteControl (
    stateInfo As StateInfo,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    timeout As Integer
) As AdsErrorCode
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598] New ADS status and device status.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be sent to the ADS device
offset	Type: System.Int32 Offset of the data in the stream.
length	Type: System.Int32 Length of the data in the stream.
timeout	Type: System.Int32 The timeout.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

TryWriteControl Overload [▶ 269]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.29 AdsConnection.Write Method

Overload List

	Name	Description
	Write(Int32, AdsStream) [▶ 274]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32) [▶ 274]	Trigger Client Method/Command.
	Write(UInt32, UInt32, Int32) [▶ 275]	Trigger Client Method/Command.
	Write(UInt32, UInt32, AdsStream) [▶ 276]	Writes data synchronously to an ADS device.
	Write(Int32, AdsStream, Int32, Int32) [▶ 276]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream, Int32) [▶ 277]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, .Byte., Int32, Int32) [▶ 278]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 279]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, .Byte., Int32, Int32, Int32) [▶ 280]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream, Int32, Int32, Int32) [▶ 281]	Writes data synchronously to an ADS device.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Write Method (Int32, AdsStream)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    int variableHandle,
    AdsStream dataStream
)
```

VB

```
Public Sub Write (
    variableHandle As Integer,
    dataStream As AdsStream
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.

Implements

IAdsHandleAccess.Write(Int32, AdsStream) [▶ 494]

Reference

AdsConnection Class [▶ 148]

Write Overload [▶ 273]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Write Method (UInt32, UInt32)

Trigger Client Method/Command.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    uint indexGroup,
    uint indexOffset
)
```

VB

```
Public Sub Write (
    indexGroup As UInteger,
    indexOffset As UInteger
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.

Remarks

This method is used to trigger Client Methods/Commands without parameters.

Reference

AdsConnection Class [▶ 148]

Write Overload [▶ 273]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Write Method (UInt32, UInt32, Int32)

Trigger Client Method/Command.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    uint indexGroup,
    uint indexOffset,
    int timeout
)
```

VB

```
Public Sub Write (
    indexGroup As UInteger,
    indexOffset As UInteger,
    timeout As Integer
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
timeout	Type: System.Int32 The timeout.

Remarks

This method is used to trigger Client Methods/Commands without parameters.

Reference

AdsConnection Class [▶ 148]

Write Overload [▶ 273]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Write Method (UInt32, UInt32, AdsStream)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream
)
```

VB

```
Public Sub Write (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.

Reference

AdsConnection Class [▶ 148]

Write Overload [▶ 273]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Write Method (Int32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    int variableHandle,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Sub Write (
    variableHandle As Integer,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Implements

IAdsHandleAccess.Write(Int32, AdsStream, Int32, Int32) [▶ 495]

Reference

AdsConnection Class [▶ 148]

Write Overload [▶ 273]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Write Method (UInt32, UInt32, AdsStream, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int timeout
)
```

VB

```
Public Sub Write (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    timeout As Integer
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
timeout	Type: System.Int32 The timeout.

Reference

AdsConnection Class [▶ 148]

Write Overload [▶ 273]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Write Method (UInt32, UInt32, .Byte., Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    uint indexGroup,
    uint indexOffset,
    byte[] writeBuffer,
    int offset,
    int length
)
```

VB

```
Public Sub Write (
    indexGroup As UInteger,
    indexOffset As UInteger,
    writeBuffer As Byte(),
    offset As Integer,
    length As Integer
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.

writeBuffer	Type: .System.Byte. The write buffer.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

Write Overload [▶ 273]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Write Method (UInt32, UInt32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Sub Write (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Reference

AdsConnection Class [▶ 148]

Write Overload [▶ 273]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Write Method (UInt32, UInt32, .Byte., Int32, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    uint indexGroup,
    uint indexOffset,
    byte[] writeBuffer,
    int offset,
    int length,
    int timeout
)
```

VB

```
Public Sub Write (
    indexGroup As UInteger,
    indexOffset As UInteger,
    writeBuffer As Byte(),
    offset As Integer,
    length As Integer,
    timeout As Integer
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
writeBuffer	Type: .System.Byte. The write buffer.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
timeout	Type: System.Int32 The timeout.

Reference

AdsConnection Class [▶ 148]

Write Overload [▶ 273]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.Write Method (UInt32, UInt32, AdsStream, Int32, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length,
    int timeout
)
```

VB

```
Public Sub Write (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    timeout As Integer
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
timeout	Type: System.Int32 The timeout.

Reference

AdsConnection Class [▶ 148]

Write Overload [▶ 273]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.30 AdsConnection.WriteAny Method

Overload List

	Name	Description
	WriteAny(Int32, Object) [▶ 282]	Writes an object synchronously to an ADS device.

	Name	Description
	WriteAny(Int32, Object, .Int32.) [▶ 283]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.
	WriteAny(UInt32, UInt32, Object) [▶ 283]	Writes an object synchronously to an ADS device.
	WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 284]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.
	WriteAny(UInt32, UInt32, Object, .Int32., Int32) [▶ 285]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.WriteAny Method (Int32, Object)

Writes an object synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteAny(
    int variableHandle,
    Object value
)
```

VB

```
Public Sub WriteAny (
    variableHandle As Integer,
    value As Object
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable.
value	Type: System.Object Object to write to the ADS device.

Implements

IAdsAnyAccess.WriteAny(Int32, Object) [▶ 466]

Reference

AdsConnection Class [▶ 148]

WriteAny Overload [▶ 281]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.WriteAny Method (Int32, Object, .Int32.)

Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteAny(
    int variableHandle,
    Object value,
    int[] args
)
```

VB

```
Public Sub WriteAny (
    variableHandle As Integer,
    value As Object,
    args As Integer()
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable.
value	Type: System.Object Object to write to the ADS device.
args	Type: .System.Int32. Additional arguments.

Implements

IAdsAnyAccess.WriteAny(Int32, Object, .Int32.) [▶ 467]

Reference

AdsConnection Class [▶ 148]

WriteAny Overload [▶ 281]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.WriteAny Method (UInt32, UInt32, Object)

Writes an object synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteAny(
    uint indexGroup,
    uint indexOffset,
    Object value
)
```

VB

```
Public Sub WriteAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    value As Object
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
value	Type: System.Object Object to write to the ADS device.

Implements

IAdsAnyAccess.WriteAny(UInt32, UInt32, Object) [▶ 468]

Reference

AdsConnection Class [▶ 148]

WriteAny Overload [▶ 281]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.WriteAny Method (UInt32, UInt32, Object, .Int32.)

Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteAny(
    uint indexGroup,
    uint indexOffset,
    Object value,
    int[] args
)
```

VB

```
Public Sub WriteAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    value As Object,
    args As Integer()
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
value	Type: System.Object Object to write to the ADS device.
args	Type: .System.Int32. Additional arguments.

Implements

IAdsAnyAccess.WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 468]

Reference

AdsConnection Class [▶ 148]

WriteAny Overload [▶ 281]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.WriteAny Method (UInt32, UInt32, Object, .Int32., Int32)

Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteAny(
    uint indexGroup,
    uint indexOffset,
    Object value,
    int[] args,
    int timeout
)
```

VB

```
Public Sub WriteAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    value As Object,
    args As Integer(),
    timeout As Integer
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
value	Type: System.Object Object to write to the ADS device.
args	Type: .System.Int32. Additional arguments.
timeout	Type: System.Int32 The timeout.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

WriteAny Overload [▶ 281]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.31 AdsConnection.WriteControl Method

Overload List

	Name	Description
	WriteControl(StateInfo) [▶ 287]	Changes the ADS status and the device status of an ADS server.
	WriteControl(StateInfo, Int32) [▶ 287]	Changes the ADS status and the device status of an ADS server.
	WriteControl(StateInfo, AdsStream, Int32, Int32) [▶ 288]	Changes the ADS status and the device status of an ADS server.
	WriteControl(StateInfo, AdsStream, Int32, Int32, Int32) [▶ 289]	Changes the ADS status and the device status of an ADS server.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.WriteControl Method (StateInfo)

Changes the ADS status and the device status of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteControl(
    StateInfo stateInfo
)
```

VB

```
Public Sub WriteControl (
    stateInfo As StateInfo
)
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598] New ADS status and device status.

Reference

AdsConnection Class [▶ 148]

WriteControl Overload [▶ 286]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.WriteControl Method (StateInfo, Int32)

Changes the ADS status and the device status of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteControl(
    StateInfo stateInfo,
    int timeout
)
```

VB

```
Public Sub WriteControl (
    stateInfo As StateInfo,
    timeout As Integer
)
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598] New ADS status and device status.
timeout	Type: System.Int32 The timeout.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

WriteControl Overload [▶ 286]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.WriteControl Method (StateInfo, AdsStream, Int32, Int32)

Changes the ADS status and the device status of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteControl(
    StateInfo stateInfo,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Sub WriteControl (
    stateInfo As StateInfo,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
)
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598] New ADS status and device status.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be sent to the ADS device
offset	Type: System.Int32 Offset of the data in the stream.
length	Type: System.Int32 Length of the data in the stream.

Reference

AdsConnection Class [▶ 148]

WriteControl Overload [▶ 286]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.WriteControl Method (StateInfo, AdsStream, Int32, Int32, Int32)

Changes the ADS status and the device status of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteControl(
    StateInfo stateInfo,
    AdsStream dataStream,
    int offset,
    int length,
    int timeout
)
```

VB

```
Public Sub WriteControl (
    stateInfo As StateInfo,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    timeout As Integer
)
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598] New ADS status and device status.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be sent to the ADS device
offset	Type: System.Int32 Offset of the data in the stream.
length	Type: System.Int32 Length of the data in the stream.
timeout	Type: System.Int32 The timeout.

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

AdsConnection Class [▶ 148]

WriteControl Overload [▶ 286]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.2.32 AdsConnection.WriteSymbol Method

Overload List

	Name	Description
	WriteSymbol(ITcAdsSymbol, Object) [▶ 290]	Writes a value to the symbol. Strings and all primitive data types(UInt32, Int32, Bool etc.) are supported. Array and structures are not supported. If a string is passed as parameter, the method attempts to parse the string according to the ADS data type of the symbol.
	WriteSymbol(String, Object, Boolean) [▶ 291]	Writes the passed object value to the specified ADS symbol.The parameter type must have the same layout as the ADS symbol.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.WriteSymbol Method (ITcAdsSymbol, Object)

Writes a value to the symbol. Strings and all primitive data types(UInt32, Int32, Bool etc.) are supported. Array and structures are not supported. If a string is passed as parameter, the method attempts to parse the string according to the ADS data type of the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteSymbol(
    ITcAdsSymbol symbol,
    Object val
)
```

VB

```
Public Sub WriteSymbol (
    symbol As ITcAdsSymbol,
    val As Object
)
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol the value is written to.
val	Type: System.Object The value to write.

Reference

AdsConnection Class [▶ 148]

WriteSymbol Overload [▶ 290]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.WriteSymbol Method (String, Object, Boolean)

Writes the passed object value to the specified ADS symbol.The parameter type must have the same layout as the ADS symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteSymbol(
    string name,
    Object value,
    bool reloadSymbolInfo
)
```

VB

```
Public Sub WriteSymbol (
    name As String,
    value As Object,
    reloadSymbolInfo As Boolean
)
```

Parameters

name	Type: System.String Name of the ADS symbol.
value	Type: System.Object Object holding the value to be written to the ADS symbol
reloadSymbolInfo	Type: System.Boolean If reload is true previously stored symbol information is cleared. As a consequence the symbol information must be obtained from the ADS server again.

Reference

AdsConnection Class [▶ 148]

WriteSymbol Overload [▶ 290]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.3 AdsConnection Events

The AdsConnection [▶ 148] type exposes the following members.

Events

	Name	Description
⚡	AdsNotification [▶ 292]	Occurs when the ADS device sends a notification to the client.
⚡	AdsNotificationError [▶ 292]	Occurs when a exception has occurred during notification management.
⚡	AdsNotificationEx [▶ 293]	Occurs when the ADS devices sends a notification to the client.
⚡	AdsStateChanged [▶ 293]	Occurs when ADS State has been changed.

	Name	Description
	AdsSymbolVersionChanged [▶ 294]	Occurs when the symbol version has been changed.
	AmsRouterNotification [▶ 294]	Occurs when the Route sends an Notification.
	ConnectionStateChanged [▶ 295]	Occurs when connection status of the AdsConnection [▶ 148] has been changed.

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.3.1 AdsConnection.AdsNotification Event

Occurs when the ADS device sends a notification to the client.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event AdsNotificationEventHandler AdsNotification
```

VB

```
Public Event AdsNotification As AdsNotificationEventHandler
```

Value

Type: TwinCAT.Ads.AdsNotificationEventHandler [▶ 342]

Implements

IAdsNotifications.AdsNotification [▶ 514]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.3.2 AdsConnection.AdsNotificationError Event

Occurs when a exception has occurred during notification management.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event AdsNotificationErrorEventHandler AdsNotificationError
```

VB

```
Public Event AdsNotificationError As AdsNotificationErrorEventHandler
```

Value

Type: TwinCAT.Ads.AdsNotificationErrorEventHandler [▶ 335]

Implements

IAdsNotifications.AdsNotificationError [▶ 515]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.3.3 AdsConnection.AdsNotificationEx Event

Occurs when the ADS devices sends a notification to the client.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event AdsNotificationExEventHandler AdsNotificationEx
```

VB

```
Public Event AdsNotificationEx As AdsNotificationExEventHandler
```

Value

Type: TwinCAT.Ads.AdsNotificationExEventHandler [▶ 346]

Implements

IAdsNotifications.AdsNotificationEx [▶ 516]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.3.4 AdsConnection.AdsStateChanged Event

Occurs when ADS State has been changed.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event AdsStateChangedEventHandler AdsStateChanged
```

VB

```
Public Event AdsStateChanged As AdsStateChangedEventHandler
```

Value

Type: TwinCAT.Ads.AdsStateChangedEventHandler [▶ 374]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.3.5 AdsConnection.AdsSymbolVersionChanged Event

Occurs when the symbol version has been changed.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event EventHandler AdsSymbolVersionChanged
```

VB

```
Public Event AdsSymbolVersionChanged As EventHandler
```

Value

Type: System.EventHandler

Remarks

This is the case when the connected ADS server restarts. This invalidates all actual opened symbol handles. The SymbolVersion counter doesn't trigger, when an online change is made on the PLC (ports 801, ..., 851 ...)

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.3.6 AdsConnection.AmsRouterNotification Event

Occurs when the Route sends an Notification.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event AmsRouterNotificationEventHandler AmsRouterNotification
```

VB

```
Public Event AmsRouterNotification As AmsRouterNotificationEventHandler
```

Value

Type: TwinCAT.Ads.AmsRouterNotificationEventHandler [▶ 454]

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

5.2.5.3.7 AdsConnection.ConnectionStateChanged Event

Occurs when connection status of the AdsConnection [▶ 148] has been changed.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event EventHandler<ConnectionStateChangedEventArgs> ConnectionStateChanged
```

VB

```
Public Event ConnectionStateChanged As EventHandler(Of ConnectionStateChangedEventArgs)
```

Value

Type: System.EventHandler.ConnectionStateChangedEventArgs [▶ 39].

Implements

IConnectionStateProvider.ConnectionStateChanged [▶ 54]

Remarks

The Connection state changes only if the IConnection [▶ 46] is established / shut down or active communication is triggered by the User of the IConnection [▶ 46] object.

Examples

The following sample shows how to keep the ConnectionState [▶ 165] updated by triggering ADS Communication.

Trigger ConnectionState changes in WPF Applications

```
private DispatcherTimer _timer = null;
private AdsSession _session = null;
//private AdsConnection _connection = null;

private void Window_Loaded(object sender, RoutedEventArgs e)
{
    _session = new AdsSession(AmsNetId.Local, 10000);
    IConnection connection = _session.Connect();
    tbConnectionState.Text = connection.ConnectionState.ToString();
    _session.ConnectionStateChanged += _session_ConnectionStateChanged;

    _timer = new DispatcherTimer();
    _timer.Interval = TimeSpan.FromMilliseconds(200);
    _timer.Tick += TimerOnTick;

    _timer.Start();
}

private void Window_Unloaded(object sender, RoutedEventArgs e)
{
    _timer.Stop();
    _session.Dispose();
}

private void _session_ConnectionStateChanged(object sender, TwinCAT.ConnectionStateChangedEventArgs
e)
{
    // ConnectionStateChanged will be triggered by communication Invokes
    tbConnectionState.Text = e.NewState.ToString();
}

private void TimerOnTick(object sender, EventArgs eventArgs)
{
    // The Timer Event will occur here in the UI thread because its an DispatcherTimer event!
    // An active ADS request will trigger Connection State periodically!
    StateInfo stateInfo;
    if (_session.Connection.TryReadState(out stateInfo) == AdsErrorCode.NoError)
    {
    tbAdsState.Text = stateInfo.AdsState.ToString();
    }
    else
    {
    tbAdsState.Text = "Invalid";
    }
}
```

Reference

AdsConnection Class [▶ 148]

TwinCAT.Ads Namespace [▶ 108]

AdsConnection.ConnectionState [▶ 165]

5.2.6 AdsDatatypeArrayInfo Class

Array definition for a single dimension.

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.AdsDatatypeArrayInfo

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class AdsDatatypeArrayInfo
```

VB

```
Public Class AdsDatatypeArrayInfo
```

The AdsDatatypeArrayInfo type exposes the following members.

Properties

	Name	Description
	Elements [▶ 298]	Gets the number of elements.
	LowerBound [▶ 298]	Gets the lower bound.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.6.1 AdsDatatypeArrayInfo Properties

The AdsDatatypeArrayInfo [▶ 296] type exposes the following members.

Properties

	Name	Description
	Elements [▶ 298]	Gets the number of elements.
	LowerBound [▶ 298]	Gets the lower bound.

Reference

AdsDatatypeArrayInfo Class [▶ 296]

TwinCAT.Ads Namespace [▶ 108]

5.2.6.1.1 AdsDatatypeArrayInfo.Elements Property

Gets the number of elements.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Elements { get; }
```

VB

```
Public ReadOnly Property Elements As Integer
    Get
```

Property Value

Type: Int32

Reference

AdsDatatypeArrayInfo Class [▶ 296]

TwinCAT.Ads Namespace [▶ 108]

5.2.6.1.2 AdsDatatypeArrayInfo.LowerBound Property

Gets the lower bound.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int LowerBound { get; }
```

VB

```
Public ReadOnly Property LowerBound As Integer
    Get
```

Property Value

Type: Int32

Reference

AdsDatatypeArrayInfo Class [▶ 296]

TwinCAT.Ads Namespace [▶ 108]

5.2.6.2 AdsDatatypeArrayInfo Methods

The AdsDatatypeArrayInfo [▶ 296] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

AdsDatatypeArrayInfo Class [▶ 296]

TwinCAT.Ads Namespace [▶ 108]

5.2.7 AdsDatatypeId Enumeration

ADS data types.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public enum AdsDatatypeId
```

VB

```
Public Enumeration AdsDatatypeId
```

Members

	Member name	Value	Description
	ADST_VOID	0	Empty Type
	ADST_INT8	16	Integer 8 Bit
	ADST_UINT8	17	Unsigned integer 8 Bit
	ADST_INT16	2	Integer 16 Bit
	ADST_UINT16	18	Unsigned integer 16 Bit
	ADST_INT32	3	Integer 32 Bit
	ADST_UINT32	19	Unsigned Integer 32 Bit
	ADST_INT64	20	LONG Integer 64 Bit
	ADST_UINT64	21	Unsigned Long integer 64 Bit
	ADST_REAL32	4	Real (32 Bit)
	ADST_REAL64	5	Real 64 Bit
	ADST_BIGTYPE	65	Blob
	ADST_STRING	30	STRING
	ADST_WSTRING	31	WSTRING

	Member name	Value	Description
	ADST_REAL80	32	ADS REAL80
	ADST_BIT	33	ADS BIT
	ADST_MAXTYPES	34	Internal Only

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.8 AdsDatatypeNotSupportedException Class

The exception that is thrown when a ADS datatype is not supported.

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.Ads.AdsDatatypeNotSupportedException

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class AdsDatatypeNotSupportedException : AdsException
```

VB

```
<SerializableAttribute>
Public Class AdsDatatypeNotSupportedException
    Inherits AdsException
```

The AdsDatatypeNotSupportedException type exposes the following members.

Constructors

	Name	Description
	AdsDatatypeNotSupportedException. [▶ 302]	Initializes a new Instance of the AdsDatatypeNotSupportedException class.
	AdsDatatypeNotSupportedException(String) [▶ 302]	Initializes a new Instance of the AdsDatatypeNotSupportedException class.
	AdsDatatypeNotSupportedException(String, Exception) [▶ 303]	Initializes a new Instance of the AdsDatatypeNotSupportedException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.8.1 AdsDatatypeNotSupportedException Constructor

Overload List

	Name	Description
	AdsDatatypeNotSupportedException. [▶ 302]	Initializes a new Instance of the AdsDatatypeNotSupportedException class.
	AdsDatatypeNotSupportedException(String) [▶ 302]	Initializes a new Instance of the AdsDatatypeNotSupportedException class.
	AdsDatatypeNotSupportedException(String, Exception) [▶ 303]	Initializes a new Instance of the AdsDatatypeNotSupportedException class.

Reference

AdsDatatypeNotSupportedException Class [▶ 300]

TwinCAT.Ads Namespace [▶ 108]

5.2.8.1.1 AdsDatatypeNotSupportedException Constructor

Initializes a new Instance of the AdsDatatypeNotSupportedException class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsDatatypeNotSupportedException()
```

VB

```
Public Sub New
```

Reference

AdsDatatypeNotSupportedException Class [▶ 300]

AdsDatatypeNotSupportedException Overload [▶ 302]

TwinCAT.Ads Namespace [▶ 108]

5.2.8.1.2 AdsDatatypeNotSupportedException Constructor (String)

Initializes a new Instance of the AdsDatatypeNotSupportedException class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsDatatypeNotSupportedException(
    string message
)
```

VB

```
Public Sub New (
    message As String
)
```

Parameters

message	Type: System.String The message.

Reference

AdsDatatypeNotSupportedException Class [▶ 300]

AdsDatatypeNotSupportedException Overload [▶ 302]

TwinCAT.Ads Namespace [▶ 108]

5.2.8.1.3 AdsDatatypeNotSupportedException Constructor (String, Exception)

Initializes a new Instance of the AdsDatatypeNotSupportedException class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsDatatypeNotSupportedException(
    string message,
    Exception innerException
)
```

VB

```
Public Sub New (
    message As String,
    innerException As Exception
)
```

Parameters

message	Type: System.String The message.
innerException	Type: System.Exception The inner exception.

Reference

AdsDatatypeNotSupportedException Class [▶ 300]

AdsDatatypeNotSupportedException Overload [▶ 302]

TwinCAT.Ads Namespace [▶ 108]

5.2.8.2 AdsDatatypeNotSupportedException Properties

The AdsDatatypeNotSupportedException [▶ 300] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

AdsDatatypeNotSupportedException Class [▶ 300]

TwinCAT.Ads Namespace [▶ 108]

5.2.8.3 AdsDatatypeNotSupportedException Methods

The AdsDatatypeNotSupportedException [▶ 300] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

AdsDatatypeNotSupportedException Class [▶ 300]

TwinCAT.Ads Namespace [▶ 108]

5.2.8.4 AdsDatatypeNotSupportedException Events

The AdsDatatypeNotSupportedException [▶ 300] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

AdsDatatypeNotSupportedException Class [▶ 300]

TwinCAT.Ads Namespace [▶ 108]

5.2.9 AdsErrorCode Enumeration

Describes the ADS error that occured.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public enum AdsErrorCode
```

VB

```
Public Enumeration AdsErrorCode
```

Members

	Member name	Value	Description
	NoError	0	No Error. Error code: 0(0x000).
	InternalError	1	Internal Error. Error code: 1(0x001).
	NoRTime	2	No Rtime. Error code: 2(0x002).
	LockedMemoryError	3	Allocation locked memory error. Error code: 3(0x003).
	MailBoxError	4	Insert mailbox error. Error code: 4(0x004).
	WrongHMsg	5	Wrong receive HMSG. Error code: 5(0x005).

	Member name	Value	Description
	TargetPortNotFound	6	Target port not found. Error code: 6(0x006).
	TargetMachineNotFound	7	Target machine not found. Error code: 7(0x007).
	UnknownCommandID	8	Unknown command ID. Error code: 8(0x008).
	BadTaskID	9	Bad task ID. Error code: 9(0x009).
	NoIO	10	No IO. Error code: 10(0x00A).
	UnknwonAmsCommand	11	Unknown AMS command. Error code: 11(0x00B).
	Win32Error	12	Win 32 error. Error code: 12(0x00C).
	PortNotConnected	13	Port is not connected. Error code: 13(0x00D).
	InvalidAmsLength	14	Invalid AMS length. Error code: 14(0x00E).
	InvalidAmsNetID	15	Invalid AMS Net ID. Error code: 15(0x00F).
	LowInstallLevel	16	Low Installation level. Error code: 16(0x010).
	NoDebug	17	No debug available. Error code: 17(0x011).
	PortDisabled	18	Port disabled. Error code: 18(0x012).
	PortConnected	19	Port is already connected. Error code: 19(0x013).
	AmsSyncWin32Error	20	AMS Sync Win32 error. Error code: 20(0x014).
	SyncTimeOut	21	AMS Sync timeout. Error code: 21(0x015).
	AmsSyncAmsError	22	AMS Sync AMS error Error code: 22(0x016).
	AmsSyncNoIndexMap	23	AMS Sync no index map. Error code: 23(0x017).
	InvalidAmsPort	24	Invalid AMS port. Error code: 24(0x018).
	NoMemory	25	No memory. Error code: 25(0x019).
	TCPSendError	26	TCP send error. Error code: 26(0x01A).
	HostUnreachable	27	Host unreachable. Error code: 27(0x1B).
	AmsInvalidFragment	28	Invalid AMS fragment. Error code: 28(0x1C).
	NoLockedMemory	1280	Router: no locked memory.

	Member name	Value	Description
			Error code: 1280(0x500).
	ResizeMemory	1281	Router: The size of the router memory could not be changed. Error code: 1281(0x501).
	MailboxFull	1282	Router: mailbox full. Error code: 1282(0x502).
	DebugBoxFull	1283	Router: The mailbox has reached the maximum number of possible messages. Error code: 1283(0x503).
	UnknownPortType	1284	Router: Unknown Port Type Error code: 1284(0x504).
	RouterNotInitialized	1285	Router: Router is not initialized. Error code: 1285(0x505).
	PortAlreadyInUse	1286	Router: The desired port number is already assigned. Error code: 1286(0x506).
	PortNotRegistered	1287	Router: Port not registered. Error code: 1287(0x507).
	NoMoreQueues	1288	Router: The maximum number of Ports reached. Error code: 1288(0x508).
	InvalidPort	1289	Router: The port is invalid. Error code: 1289(0x509).
	RouterNotActive	1290	Router: TwinCAT Router not active. Error code: 1290(0x50A).
	DeviceError	1792	error class <device error"> Error code: 1792(0x700).
	DeviceServiceNotSupported	1793	Service is not supported by server. Error code: 1793(0x701).
	DeviceInvalidGroup	1794	Invalid index group. Error code: 1794(0x702).
	DeviceInvalidOffset	1795	Invalid index offset. Error code: 1795(0x703).
	DeviceInvalidAccess	1796	Reading/writing not permitted. Error code: 1796(0x704).
	DeviceInvalidSize	1797	Parameter size not correct. Error code: 1797(0x705).
	DeviceInvalidData	1798	Invalid parameter value(s). Error code: 1798(0x706).
	DeviceNotReady	1799	Device is not in a ready state. Error code: 1799(0x707).
	DeviceBusy	1800	Device is busy. Error code: 1800(0x708).
	DeviceInvalidContext	1801	Invalid context (must be in Windows). Error code: 1801(0x709).
	DeviceNoMemory	1802	Out of memory. Error code: 1802(0x70a).

	Member name	Value	Description
	DeviceInvalidParam	1803	Invalid parameter value(s). Error code: 1803(0x70b).
	DeviceNotFound	1804	Not found(files, ...). Error code: 1804(0x70c).
	DeviceSyntaxError	1805	Syntax error in command or file. Error code: 1805(0x70d).
	DeviceIncompatible	1806	Objects do not match. Error code: 1806(0x70e).
	DeviceExists	1807	Object already exists. Error code: 1807(0x70f).
	DeviceSymbolNotFound	1808	Symbol not found. Error code: 1808(0x7010).
	DeviceSymbolVersionInvalid	1809	Symbol version is invalid. Error code: 1809(0x711).
	DeviceInvalidState	1810	Server is not i a valid state. Error code: 1810(0x712).
	DeviceTransModeNotSupported	1811	ADS transmode is not supported. Error code: 1811(0x713).
	DeviceNotifyHandleInvalid	1812	Notification handle is invalid. Error code: 1812(0x714).
	DeviceClientUnknown	1813	Notification vlient not registered. Error code: 1813(0x715).
	DeviceNoMoreHandles	1814	No more notification handles. Error code: 1814(0x716).
	DeviceInvalidWatchsize	1815	Size for watch to big. Error code: 1815(0x717).
	DeviceNotInitialized	1816	Device is not initialized. Errocr code: 1818(0x718).
	DeviceTimeOut	1817	Devicee has a timeout. Error code: 1817(0x719).
	DeviceNoInterface	1818	Query interface has failed. Error code: 1818(0x71A).
	DeviceInvalidInterface	1819	Wrong interface required. Error code: 1819(0x71B).
	DeviceInvalidCLSID	1820	Class ID is invalid. Error code: 1820(0x71C).
	DeviceInvalidObjectID	1821	Object ID is invalid. Error code: 1821(0x71D).
	DeviceRequestIsPending	1822	Device: Request is Pending. Error code: 1822(0x71E).
	DeviceRequestIsAborted	1823	Device: Request is Aborted. Error code: 1823(0x71F).
	DeviceSignalWarning	1824	Device: Signal warning. Error code: 1824(0x720).
	DeviceInvalidArrayIndex	1825	Device: Invalid Array Index (ADSERR_DEVICE_INVALIDARRAYIDX)

	Member name	Value	Description
			Error code: 1825(0x721).
	DeviceSymbolNotActive	1826	Device: Symbol not Active Error code: 1826(0x722).
	DeviceAccessDenied	1827	Device: Access denied. Error code: 1827(0x723).
	DeviceLicenseNotFound	1828	Device: license not found Error code: 1828(0x724).
	DeviceLicenseExpired	1829	Device: license expired Error code: 1829(0x725).
	DeviceLicenseExceeded	1830	Device: license exceeded Error code: 1830(0x726).
	DeviceLicenseInvalid	1831	Device: license invalid Error code: 1831(0x727).
	DeviceLicenseSystemId	1832	Device: license invalid system id Error code: 1832(0x728).
	DeviceLicenseNoTimeLimit	1833	Device: license not time limited Error code: 1833(0x729).
	DeviceLicenseFutureIssue	1834	Device: license issue time in the future Error code: 1834(0x72A).
	DeviceLicenseTimeToLong	1835	Device: license time period to long Error code: 1835(0x72B).
	DeviceException	1836	Device: Exception in device specific code Error code: 1836(0x72C).
	DeviceLicenseDuplicated	1837	Device: license file read twice Error code: 1837(0x72D).
	DeviceSignatureInvalid	1838	Device: invalid signature Error code: 1838(0x72E).
	DeviceCertificateInvalid	1839	Device: public key certificate Error code: 1839(0x72F).
	DeviceLicenseOemNotFound	1840	Device: public key of OEM unknown Error code: 1840(0x730).
	DeviceLicenseRestricted	1841	Device: license not valid for this system id type Error code: 1841(0x731).
	DeviceLicenseDemoDenied	1842	Device: trial license denied Error code: 1842(0x732).
	DeviceInvalidFncId	1843	Device: function id is invalid Error code: 1843(0x733).
	DeviceOutOfRange	1844	Device: a parameter, an index, an iterator, ... is out of range Error code: 1844(0x734).
	DeviceInvalidAlignment	1845	Device: invalid alignment Error code: 1845(0x735).
	DeviceLicensePlatform	1846	Device: license invalid platform level Error code: 1846(0x736).
	ClientError	1856	Error class <client error>

	Member name	Value	Description
			Error code: 1856(0x740).
	ClientInvalidParameter	1857	Parameter at service is invalid. Error code: 1857(0x741).
	ClientListEmpty	1858	Polling list is empty. Error code: 1858(0x742).
	ClientVariableInUse	1859	Variable connection is already in use. Error code: 1859(0x743).
	ClientDuplicateInvokeID	1860	Invoke ID already in use. Error code: 1860(0x744).
	ClientSyncTimeOut	1861	Timeout has elapsed. Error code: 1861(x745).
	ClientW32OR	1862	Error in win32 subsystem. Error code: 1862(0x746).
	ClientTimeoutInvalid	1863	Timeout value is invalid. Error code: 1863(0x747).
	ClientPortNotOpen	1864	ADS port is not opened. Error code: 1864(0x748).
	ClientNoAmsAddr	1865	No AMS Address. Error code: 1865(0x749).
	ClientSyncInternal	1872	An internal in ADS sync has occurred. Error code: 1872(0x750).
	ClientAddHash	1873	Hash table overflow. Error code: 1873(0x751).
	ClientRemoveHash	1874	There are no more symbols in the hash table. Error code: 1874(0x752).
	ClientNoMoreSymbols	1875	There are no more symbols in cache. Error code: 1875(0x753).
	ClientSyncResInvalid	1876	An invalid response has been received. Error code: 1876(0x754).
	ClientSyncPortLocked	1877	Sync port is locked. Error code: 1877(0x755).
	ClientQueueFull	32768	Client queue is full
	WSA_ConnRefused	10061	Windows sockets connection refused (0x274d, 10061)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.10 AdsErrorException Class

The exception that is thrown when an ADS error occurs.

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.Ads.AdsErrorException
          TwinCAT.Ads.AdsSumCommandException [▶ 384]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class AdsErrorException : AdsException
```

VB

```
<SerializableAttribute>
Public Class AdsErrorException
    Inherits AdsException
```

The AdsErrorException type exposes the following members.

Constructors

	Name	Description
	AdsErrorException. [▶ 313]	Initializes a new Instance of the AdsErrorException class.
	AdsErrorException(String, AdsErrorCode) [▶ 313]	Initializes a new Instance of the AdsErrorException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	ErrorCode [▶ 314]	Gets the error code of the Exception.
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Create(AdsErrorCode) [▶ 316]	Creates the AdsErrorException
	Create(String, AdsErrorCode) [▶ 316]	Creates the AdsErrorException
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData [▶ 317]	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Overrides Exception.GetObjectData(SerializationInfo, StreamingContext).)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.10.1 AdsErrorException Constructor

Overload List

	Name	Description
	AdsErrorException. [▶ 313]	Initializes a new Instance of the AdsErrorException class.
	AdsErrorException(String, AdsErrorCode) [▶ 313]	Initializes a new Instance of the AdsErrorException class.

Reference

AdsErrorException Class [▶ 310]

TwinCAT.Ads Namespace [▶ 108]

5.2.10.1.1 AdsErrorException Constructor

Initializes a new Instance of the AdsErrorException class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorException()
```

VB

```
Public Sub New
```

Reference

AdsErrorException Class [▶ 310]

AdsErrorException Overload [▶ 312]

TwinCAT.Ads Namespace [▶ 108]

5.2.10.1.2 AdsErrorException Constructor (String, AdsErrorCode)

Initializes a new Instance of the AdsErrorException class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorException(
    string message,
    AdsErrorCode errorCode
)
```

VB

```
Public Sub New (
    message As String,
    errorCode As AdsErrorCode
)
```

Parameters

message	Type: System.String The message.
errorCode	Type: TwinCAT.Ads.AdsErrorCode [▶ 305] The error code.

Reference

AdsErrorException Class [▶ 310]

AdsErrorException Overload [▶ 312]

TwinCAT.Ads Namespace [▶ 108]

5.2.10.2 AdsErrorException Properties

The AdsErrorException [▶ 310] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	ErrorCode [▶ 314]	Gets the error code of the Exception.
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

AdsErrorException Class [▶ 310]

TwinCAT.Ads Namespace [▶ 108]

5.2.10.2.1 AdsErrorException.ErrorCode Property

Gets the error code of the Exception.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode ErrorCode { get; }
```

VB

```
Public ReadOnly Property ErrorCode As AdsErrorCode
    Get
```

Property Value

Type: AdsErrorCode [▶ 305]
The error code.

Reference

AdsErrorException Class [▶ 310]

TwinCAT.Ads Namespace [▶ 108]

5.2.10.3 AdsErrorException Methods

The AdsErrorException [▶ 310] type exposes the following members.

Methods

	Name	Description
	Create(AdsErrorCode) [▶ 316]	Creates the AdsErrorException
	Create(String, AdsErrorCode) [▶ 316]	Creates the AdsErrorException
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData [▶ 317]	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Overrides Exception.GetObjectData(SerializationInfo, StreamingContext).)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

AdsErrorException Class [▶ 310]

TwinCAT.Ads Namespace [▶ 108]

5.2.10.3.1 AdsErrorException.Create Method

Overload List

	Name	Description
	Create(AdsErrorCode) [▶ 316]	Creates the AdsErrorException

	Name	Description
 	Create(String, AdsErrorCode) [▶ 316]	Creates the AdsErrorException

Reference

AdsErrorException Class [▶ 310]

TwinCAT.Ads Namespace [▶ 108]

AdsErrorException.Create Method (AdsErrorCode)

Creates the AdsErrorException

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static AdsErrorException Create(
    AdsErrorCode adsErrorCode
)
```

VB

```
Public Shared Function Create (
    adsErrorCode As AdsErrorCode
) As AdsErrorException
```

Parameters

adsErrorCode	Type: TwinCAT.Ads.AdsErrorCode [▶ 305] The ads error code.

Return Value

Type: AdsErrorException [▶ 310]
AdsErrorException.

Reference

AdsErrorException Class [▶ 310]

Create Overload [▶ 315]

TwinCAT.Ads Namespace [▶ 108]

AdsErrorException.Create Method (String, AdsErrorCode)

Creates the AdsErrorException

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static AdsErrorException Create(
    string message,
    AdsErrorCode adsErrorCode
)
```

VB

```
Public Shared Function Create (
    message As String,
    adsErrorCode As AdsErrorCode
) As AdsErrorException
```

Parameters

message	Type: System.String The message.
adsErrorCode	Type: TwinCAT.Ads.AdsErrorCode [▶ 305] The ads error code.

Return Value

Type: AdsErrorException [▶ 310]
AdsErrorException.

Reference

AdsErrorException Class [▶ 310]

Create Overload [▶ 315]

TwinCAT.Ads Namespace [▶ 108]

5.2.10.3.2 AdsErrorException.GetObjectData Method

When overridden in a derived class, sets the SerializationInfo with information about the exception.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override void GetObjectData(
    SerializationInfo info,
    StreamingContext context
)
```

VB

```
Public Overrides Sub GetObjectData (
    info As SerializationInfo,
    context As StreamingContext
)
```

Parameters

info	Type: System.Runtime.Serialization.SerializationInfo The SerializationInfo that holds the serialized object data about the exception being thrown.
context	Type: System.Runtime.Serialization.StreamingContext The StreamingContext that contains contextual information about the source or destination.

Implements

ISerializable.GetObjectData(SerializationInfo, StreamingContext)
_Exception.GetObjectData(SerializationInfo, StreamingContext)

Exceptions

Exception	Condition
ArgumentNullException	info

Reference

AdsErrorException Class [▶ 310]

TwinCAT.Ads Namespace [▶ 108]

5.2.10.4 AdsErrorException Events

The AdsErrorException [▶ 310] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

AdsErrorException Class [▶ 310]

TwinCAT.Ads Namespace [▶ 108]

5.2.11 AdsException Class

Base class for all exceptions thrown by this class.

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException
        More... [▶ 320]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class AdsException : ApplicationException
```

VB

```
<SerializableAttribute>
Public Class AdsException
    Inherits ApplicationException
```

The AdsException type exposes the following members.

Constructors

	Name	Description
	AdsException. [▶ 321]	Initializes a new Instance of the AdsException class.
	AdsException(String) [▶ 321]	Initializes a new Instance of the AdsException class.
	AdsException(String, Exception) [▶ 322]	Initializes a new Instance of the AdsException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)

	Name	Description
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

TwinCAT.Ads Namespace [▶ 108]

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException
        TwinCAT.Ads.AdsDatatypeNotSupportedException [▶ 300]
        TwinCAT.Ads.AdsErrorException [▶ 310]
        TwinCAT.Ads.AdsInitializeException [▶ 324]
        TwinCAT.Ads.AdsInvalidNotificationException [▶ 328]
        TwinCAT.Ads.AdsSymbolException [▶ 389]
        TwinCAT.Ads.SymbolException [▶ 606]
        TwinCAT.ClientNotConnectedException [▶ 36]
        TwinCAT.SessionException [▶ 90]
        TwinCAT.TypeSystem.DataTypeException [▶ 1198]
        TwinCAT.TypeSystem.MarshalException [▶ 1712]

5.2.11.1 AdsException Constructor

Overload List

	Name	Description
	AdsException. [▶ 321]	Initializes a new Instance of the AdsException class.
	AdsException(String) [▶ 321]	Initializes a new Instance of the AdsException class.
	AdsException(String, Exception) [▶ 322]	Initializes a new Instance of the AdsException class.

Reference

AdsException Class [▶ 318]

TwinCAT.Ads Namespace [▶ 108]

5.2.11.1.1 AdsException Constructor

Initializes a new Instance of the AdsException class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsException()
```

VB

```
Public Sub New
```

Reference

AdsException Class [▶ 318]

AdsException Overload [▶ 320]

TwinCAT.Ads Namespace [▶ 108]

5.2.11.1.2 AdsException Constructor (String)

Initializes a new Instance of the AdsException class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsException(
    string message
)
```

VB

```
Public Sub New (
    message As String
)
```

Parameters

message	Type: System.String A message that describes the error.

Reference

AdsException Class [▶ 318]

AdsException Overload [▶ 320]

TwinCAT.Ads Namespace [▶ 108]

5.2.11.1.3 AdsException Constructor (String, Exception)

Initializes a new Instance of the AdsException class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsException(
    string message,
    Exception innerException
)
```

VB

```
Public Sub New (
    message As String,
    innerException As Exception
)
```

Parameters

message	Type: System.String The error message that explains the reason for the exception.
innerException	Type: System.Exception The exception that is the cause of the current exception. If the innerException parameter is not a null reference, the current exception is raised in a catch block that handles the inner exception.

Reference

AdsException Class [▶ 318]

AdsException Overload [▶ 320]

TwinCAT.Ads Namespace [▶ 108]

5.2.11.2 AdsException Properties

The AdsException [▶ 318] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)

	Name	Description
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

AdsException Class [▶ 318]

TwinCAT.Ads Namespace [▶ 108]

5.2.11.3 AdsException Methods

The AdsException [▶ 318] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

AdsException Class [▶ 318]

TwinCAT.Ads Namespace [▶ 108]

5.2.11.4 AdsException Events

The AdsException [▶ 318] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

AdsException Class [▶ 318]

TwinCAT.Ads Namespace [▶ 108]

5.2.12 AdsInitializeException Class

Initializing exception (TcAdsDllCe resp. TcAdsDll.dll not found)

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.Ads.AdsInitializeException

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class AdsInitializeException : AdsException
```

VB

```
<SerializableAttribute>
Public Class AdsInitializeException
    Inherits AdsException
```

The AdsInitializeException type exposes the following members.

Constructors

	Name	Description
	AdsInitializeException(Exception) [▶ 326]	Initializes a new instance of the AdsInitializeException class.
	AdsInitializeException(String, Exception) [▶ 326]	Initializes a new instance of the AdsInitializeException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)

	Name	Description
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.12.1 AdsInitializeException Constructor

Overload List

	Name	Description
	AdsInitializeException(Exception) [▶ 326]	Initializes a new instance of the AdsInitializeException [▶ 324] class.
	AdsInitializeException(String, Exception) [▶ 326]	Initializes a new instance of the AdsInitializeException [▶ 324] class.

Reference

AdsInitializeException Class [▶ 324]

TwinCAT.Ads Namespace [▶ 108]

5.2.12.1.1 AdsInitializeException Constructor (Exception)

Initializes a new instance of the AdsInitializeException [▶ 324] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsInitializeException(
    Exception inner
)
```

VB

```
Public Sub New (
    inner As Exception
)
```

Parameters

inner	Type: System.Exception The inner.

Reference

AdsInitializeException Class [▶ 324]

AdsInitializeException Overload [▶ 326]

TwinCAT.Ads Namespace [▶ 108]

5.2.12.1.2 AdsInitializeException Constructor (String, Exception)

Initializes a new instance of the AdsInitializeException [▶ 324] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsInitializeException(
    string message,
    Exception inner
)
```

VB

```
Public Sub New (
    message As String,
    inner As Exception
)
```

Parameters

message	Type: System.String The message.
inner	Type: System.Exception The inner.

Reference

AdsInitializeException Class [▶ 324]

AdsInitializeException Overload [▶ 326]

TwinCAT.Ads Namespace [▶ 108]

5.2.12.2 AdsInitializeException Properties

The AdsInitializeException [▶ 324] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

AdsInitializeException Class [▶ 324]

TwinCAT.Ads Namespace [▶ 108]

5.2.12.3 AdsInitializeException Methods

The AdsInitializeException [▶ 324] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

AdsInitializeException Class [▶ 324]

TwinCAT.Ads Namespace [▶ 108]

5.2.12.4 AdsInitializeException Events

The AdsInitializeException [▶ 324] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

AdsInitializeException Class [▶ 324]

TwinCAT.Ads Namespace [▶ 108]

5.2.13 AdsInvalidNotificationException Class

This AdsInvalidNotificationException is created if the length of the notification data is 0. This indicates that the notification handle is not valid any more. This exception is passed to the AdsNotificationErrorEvent.

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.Ads.AdsInvalidNotificationException

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public sealed class AdsInvalidNotificationException : AdsException
```

VB

```
<SerializableAttribute>
Public NotInheritable Class AdsInvalidNotificationException
    Inherits AdsException
```

The AdsInvalidNotificationException type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	Handle [▶ 330]	Handle of the notification.
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)
	TimeStamp [▶ 331]	Gets the Time stamp as long

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)

	Name	Description
	GetObjectData [▶ 332]	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Overrides Exception.GetObjectData(SerializationInfo, StreamingContext).)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.13.1 AdsInvalidNotificationException Properties

The AdsInvalidNotificationException [▶ 328] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	Handle [▶ 330]	Handle of the notification.
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)
	TimeStamp [▶ 331]	Gets the Time stamp as long

Reference

AdsInvalidNotificationException Class [▶ 328]

TwinCAT.Ads Namespace [▶ 108]

5.2.13.1.1 AdsInvalidNotificationException.Handle Property

Handle of the notification.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Handle { get; }
```

VB

```
Public ReadOnly Property Handle As Integer
    Get
```

Property Value

Type: Int32
The handle.

Reference

AdsInvalidNotificationException Class [▶ 328]

TwinCAT.Ads Namespace [▶ 108]

5.2.13.1.2 AdsInvalidNotificationException.TimeStamp Property

Gets the Time stamp as long

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public long TimeStamp { get; }
```

VB

```
Public ReadOnly Property TimeStamp As Long
    Get
```

Property Value

Type: Int64
The time stamp.

Reference

AdsInvalidNotificationException Class [▶ 328]

TwinCAT.Ads Namespace [▶ 108]

5.2.13.2 AdsInvalidNotificationException Methods

The AdsInvalidNotificationException [▶ 328] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)

	Name	Description
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData [▶ 332]	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Overrides Exception.GetObjectData(SerializationInfo, StreamingContext).)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

AdsInvalidNotificationException Class [▶ 328]

TwinCAT.Ads Namespace [▶ 108]

5.2.13.2.1 AdsInvalidNotificationException.GetObjectData Method

When overridden in a derived class, sets the SerializationInfo with information about the exception.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override void GetObjectData(
    SerializationInfo info,
    StreamingContext context
)
```

VB

```
Public Overrides Sub GetObjectData (
    info As SerializationInfo,
    context As StreamingContext
)
```

Parameters

info	Type: System.Runtime.Serialization.SerializationInfo The SerializationInfo that holds the serialized object data about the exception being thrown.
context	Type: System.Runtime.Serialization.StreamingContext The StreamingContext that contains contextual information about the source or destination.

Implements

ISerializable.GetObjectData(SerializationInfo, StreamingContext)
_Exception.GetObjectData(SerializationInfo, StreamingContext)

Exceptions

Exception	Condition
ArgumentNullException	info

Reference

AdsInvalidNotificationException Class [▶ 328]

TwinCAT.Ads Namespace [▶ 108]

5.2.14 AdsNotificationErrorEventArgs Class

Provides data for AdsNotificationErrorEvent of the class TcAdsClient.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.Ads.AdsNotificationErrorEventArgs
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class AdsNotificationErrorEventArgs : EventArgs
```

VB

```
Public NotInheritable Class AdsNotificationErrorEventArgs
    Inherits EventArgs
```

The AdsNotificationErrorEventArgs type exposes the following members.

Constructors

	Name	Description
	AdsNotificationErrorEventArgs [▶ 334]	Initializes a new instance of the AdsNotificationErrorEventArgs class.

Properties

	Name	Description
	Exception [▶ 334]	Exception that was caught while handling notifications.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)

	Name	Description
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.14.1 AdsNotificationErrorEventArgs Constructor

Initializes a new instance of the AdsNotificationErrorEventArgs class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsNotificationErrorEventArgs(
    Exception e
)
```

VB

```
Public Sub New (
    e As Exception
)
```

Parameters

e	Type: System.Exception Exception that was caught while handling notifications.

Reference

AdsNotificationErrorEventArgs Class [▶ 333]

TwinCAT.Ads Namespace [▶ 108]

5.2.14.2 AdsNotificationErrorEventArgs Properties

The AdsNotificationErrorEventArgs [▶ 333] type exposes the following members.

Properties

	Name	Description
	Exception [▶ 334]	Exception that was caught while handling notifications.

Reference

AdsNotificationErrorEventArgs Class [▶ 333]

TwinCAT.Ads Namespace [▶ 108]

5.2.14.2.1 AdsNotificationErrorEventArgs.Exception Property

Exception that was caught while handling notifications.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Exception Exception { get; }
```

VB

```
Public ReadOnly Property Exception As Exception
    Get
```

Property Value

Type: Exception

Reference

AdsNotificationErrorEventArgs Class [▶ 333]

TwinCAT.Ads Namespace [▶ 108]

5.2.14.3 AdsNotificationErrorEventArgs Methods

The AdsNotificationErrorEventArgs [▶ 333] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

AdsNotificationErrorEventArgs Class [▶ 333]

TwinCAT.Ads Namespace [▶ 108]

5.2.15 AdsNotificationErrorEventHandler Delegate

Event handler for the AdsNotificationError event in the class TcAdsClient.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public delegate void AdsNotificationErrorEventHandler(
    Object sender,
    AdsNotificationErrorEventArgs e
)
```

VB

```
Public Delegate Sub AdsNotificationErrorEventHandler (
    sender As Object,
    e As AdsNotificationErrorEventArgs
)
```

Parameters

sender	Type: System.Object
e	Type: TwinCAT.Ads.AdsNotificationErrorEventArgs [▶ 333]

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.16 AdsNotificationEventArgs Class

Provides data for AdsNotificationEvent of the class TcAdsClient.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.Ads.AdsNotificationEventArgs
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class AdsNotificationEventArgs : EventArgs
```

VB

```
Public NotInheritable Class AdsNotificationEventArgs
    Inherits EventArgs
```

The AdsNotificationEventArgs type exposes the following members.

Constructors

	Name	Description
	AdsNotificationEventArgs [▶ 337]	Initializes a new instance of the AdsStream class AdsSyncNotificationEventArgs.

Properties

	Name	Description
	DataStream [▶ 338]	Streams that holds the notification data.
	Length [▶ 339]	Gets the Length of the data in the stream.
	NotificationHandle [▶ 339]	Gets the handle of the connection.

	Name	Description
	Offset [▶ 340]	Gets the Offset of the data in the stream.
	TimeStamp [▶ 340]	Gets the timestamp of this Notification.
	UserData [▶ 341]	Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.16.1 AdsNotificationEventArgs Constructor

Initializes a new instance of the AdsStream class AdsSyncNotificationEventArgs.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsNotificationEventArgs(
    long timeStamp,
    Object userData,
    int notificationHandle,
    int length,
    int offset,
    AdsStream dataStream
)
```

VB

```
Public Sub New (
    timeStamp As Long,
    userData As Object,
    notificationHandle As Integer,
    length As Integer,
    offset As Integer,
    dataStream As AdsStream
)
```

Parameters

timeStamp	Type: System.Int64 TwinCAT realtime timestamp.
userData	Type: System.Object An object, that is passed by AddDeviceNotification.
notificationHandle	Type: System.Int32 Handle of the connection.

length	Type: System.Int32 Length of the data in dataStream.
offset	Type: System.Int32 The offset of the data in dataStream .
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that is used to store the data.

Remarks

The TwinCAT realtime target system (even when working locally) has its own TimeSystem which is synchronized with the Desktop/User time at TwinCAT Start. From this moment on the Desktop/User time can drift from the local Realtime/Target time. can differ. The TimeStamp can be converted to a .NET DateTime Object with FromFileTimeUtc(Int64) or FromFileTime(Int64)

Reference

AdsNotificationEventArgs Class [▶ 336]

TwinCAT.Ads Namespace [▶ 108]

5.2.16.2 AdsNotificationEventArgs Properties

The AdsNotificationEventArgs [▶ 336] type exposes the following members.

Properties

	Name	Description
	DataStream [▶ 338]	Streams that holds the notification data.
	Length [▶ 339]	Gets the Length of the data in the stream.
	NotificationHandle [▶ 339]	Gets the handle of the connection.
	Offset [▶ 340]	Gets the Offset of the data in the stream.
	TimeStamp [▶ 340]	Gets the timestamp of this Notification. [▶ 336]
	UserData [▶ 341]	Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data.

Reference

AdsNotificationEventArgs Class [▶ 336]

TwinCAT.Ads Namespace [▶ 108]

5.2.16.2.1 AdsNotificationEventArgs.DataStream Property

Streams that holds the notification data.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsStream DataStream { get; }
```

VB

```
Public ReadOnly Property DataStream As AdsStream
    Get
```

Property Value

Type: AdsStream [▶ 375]

Reference

AdsNotificationEventArgs Class [▶ 336]

TwinCAT.Ads Namespace [▶ 108]

5.2.16.2.2 AdsNotificationEventArgs.Length Property

Gets the Length of the data in the stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Length { get; }
```

VB

```
Public ReadOnly Property Length As Integer
    Get
```

Property Value

Type: Int32

Reference

AdsNotificationEventArgs Class [▶ 336]

TwinCAT.Ads Namespace [▶ 108]

5.2.16.2.3 AdsNotificationEventArgs.NotificationHandle Property

Gets the handle of the connection.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int NotificationHandle { get; }
```

VB

```
Public ReadOnly Property NotificationHandle As Integer
    Get
```

Property Value

Type: Int32

Reference

AdsNotificationEventArgs Class [▶ 336]

TwinCAT.Ads Namespace [▶ 108]

5.2.16.2.4 AdsNotificationEventArgs.Offset Property

Gets the Offset of the data in the stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Offset { get; }
```

VB

```
Public ReadOnly Property Offset As Integer
    Get
```

Property Value

Type: Int32

Reference

AdsNotificationEventArgs Class [▶ 336]

TwinCAT.Ads Namespace [▶ 108]

5.2.16.2.5 AdsNotificationEventArgs.TimeStamp Property

Gets the timestamp of this Notification. [▶ 336]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public long TimeStamp { get; }
```

VB

```
Public ReadOnly Property TimeStamp As Long
    Get
```

Property Value

Type: Int64

Reference

AdsNotificationEventArgs Class [▶ 336]

TwinCAT.Ads Namespace [▶ 108]

5.2.16.2.6 AdsNotificationEventArgs.UserData Property

Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object UserData { get; }
```

VB

```
Public ReadOnly Property UserData As Object
    Get
```

Property Value

Type: Object

Reference

AdsNotificationEventArgs Class [▶ 336]

TwinCAT.Ads Namespace [▶ 108]

5.2.16.3 AdsNotificationEventArgs Methods

The AdsNotificationEventArgs [▶ 336] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

AdsNotificationEventArgs Class [▶ 336]

TwinCAT.Ads Namespace [▶ 108]

5.2.17 AdsNotificationEventHandler Delegate

Event handler for the AdsNotification event in the class TcAdsClient.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public delegate void AdsNotificationEventHandler(
    Object sender,
    AdsNotificationEventArgs e
)
```

VB

```
Public Delegate Sub AdsNotificationEventHandler (
    sender As Object,
    e As AdsNotificationEventArgs
)
```

Parameters

sender	Type: System.Object
e	Type: TwinCAT.Ads.AdsNotificationEventArgs [▶ 336]

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.18 AdsNotificationExEventArgs Class

Provides data for AdsNotificationExEvent of the class TcAdsClient.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.Ads.AdsNotificationExEventArgs
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class AdsNotificationExEventArgs : EventArgs
```

VB

```
Public NotInheritable Class AdsNotificationExEventArgs
    Inherits EventArgs
```

The AdsNotificationExEventArgs type exposes the following members.

Constructors

	Name	Description
	AdsNotificationExEventArgs [▶ 343]	Initializes a new instance of the AdsStream class AdsSyncNotificationEventArgs.

Properties

	Name	Description
	NotificationHandle [▶ 344]	Gets the handle of the connection.
	TimeStamp [▶ 345]	Gets the timestamp.
	UserData [▶ 345]	Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data.
	Value [▶ 346]	Value of the ads variable.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.18.1 AdsNotificationExEventArgs Constructor

Initializes a new instance of the AdsStream class AdsSyncNotificationEventArgs.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsNotificationExEventArgs(
    long timeStamp,
    Object userData,
    int notificationHandle,
    Object value
)
```

VB

```
Public Sub New (
    timeStamp As Long,
    userData As Object,
    notificationHandle As Integer,
    value As Object
)
```

Parameters

timeStamp	Type: System.Int64 Timestamp.
userData	Type: System.Object An object, that is passed by AddDeviceNotificationEx.
notificationHandle	Type: System.Int32 Handle of the connection.
value	Type: System.Object Value of the ADS variable.

Reference

AdsNotificationExEventArgs Class [▶ 342]

TwinCAT.Ads Namespace [▶ 108]

5.2.18.2 AdsNotificationExEventArgs Properties

The AdsNotificationExEventArgs [▶ 342] type exposes the following members.

Properties

	Name	Description
	NotificationHandle [▶ 344]	Gets the handle of the connection.
	TimeStamp [▶ 345]	Gets the timestamp.
	UserData [▶ 345]	Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data.
	Value [▶ 346]	Value of the ads variable.

Reference

AdsNotificationExEventArgs Class [▶ 342]

TwinCAT.Ads Namespace [▶ 108]

5.2.18.2.1 AdsNotificationExEventArgs.NotificationHandle Property

Gets the handle of the connection.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int NotificationHandle { get; }
```

VB

```
Public ReadOnly Property NotificationHandle As Integer
    Get
```

Property Value

Type: Int32

Reference

AdsNotificationExEventArgs Class [▶ 342]

TwinCAT.Ads Namespace [▶ 108]

5.2.18.2.2 AdsNotificationExEventArgs.TimeStamp Property

Gets the timestamp.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public long TimeStamp { get; }
```

VB

```
Public ReadOnly Property TimeStamp As Long
    Get
```

Property Value

Type: Int64

Reference

AdsNotificationExEventArgs Class [▶ 342]

TwinCAT.Ads Namespace [▶ 108]

5.2.18.2.3 AdsNotificationExEventArgs.UserData Property

Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object UserData { get; }
```

VB

```
Public ReadOnly Property UserData As Object
    Get
```

Property Value

Type: Object

Reference

AdsNotificationExEventArgs Class [▶ 342]

TwinCAT.Ads Namespace [▶ 108]

5.2.18.2.4 AdsNotificationExEventArgs.Value Property

Value of the ads variable.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object Value { get; }
```

VB

```
Public ReadOnly Property Value As Object
    Get
```

Property Value

Type: Object

Reference

AdsNotificationExEventArgs Class [▶ 342]

TwinCAT.Ads Namespace [▶ 108]

5.2.18.3 AdsNotificationExEventArgs Methods

The AdsNotificationExEventArgs [▶ 342] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

AdsNotificationExEventArgs Class [▶ 342]

TwinCAT.Ads Namespace [▶ 108]

5.2.19 AdsNotificationExEventHandler Delegate

Event handler for the AdsNotification event in the class TcAdsClient.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public delegate void AdsNotificationExEventHandler(
    Object sender,
    AdsNotificationExEventArgs e
)
```

VB

```
Public Delegate Sub AdsNotificationExEventHandler (
    sender As Object,
    e As AdsNotificationExEventArgs
)
```

Parameters

sender	Type: System.Object
e	Type: TwinCAT.Ads.AdsNotificationExEventArgs [▶ 342]

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.20 AdsSession Class

AdsSession class

Inheritance Hierarchy

System.Object
  TwinCAT.Session [▶ 67]
    TwinCAT.Ads.AdsSession
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class AdsSession : Session, IAdsSession,
    ISession, IConnectionStateProvider
```

VB

```
Public Class AdsSession
    Inherits Session
    Implements IAdsSession, ISession, IConnectionStateProvider
```

The AdsSession type exposes the following members.

Constructors

	Name	Description
	AdsSession(AmsAddress) [▶ 352]	Initializes a new instance of the AdsSession class.
	AdsSession(AmsAddress, SessionSettings) [▶ 352]	Initializes a new instance of the AdsSession class.
	AdsSession(AmsNetId, Int32) [▶ 353]	Initializes a new instance of the AdsSession class.
	AdsSession(AmsAddress, SessionSettings, Object) [▶ 354]	Initializes a new instance of the AdsSession class.
	AdsSession(AmsNetId, Int32, SessionSettings) [▶ 354]	Initializes a new instance of the AdsSession class.

Properties

	Name	Description
	Address [▶ 356]	Gets the target address of the AdsSession
	AddressSpecifier [▶ 69]	Gets the communication endpoint address string representation. (Inherited from Session [▶ 67].)
	Connection [▶ 357]	Gets the connection.
	ConnectionState [▶ 70]	Gets the current Connection state of the Session [▶ 67] (Inherited from Session [▶ 67].)
	Disposed [▶ 72]	Gets a value indicating whether this Session [▶ 67] is disposed. (Inherited from Session [▶ 67].)
	EstablishedAt [▶ 72]	Gets the UTC time when the session was established. (Inherited from Session [▶ 67].)
	Id [▶ 73]	Gets the Session Identifier (Inherited from Session [▶ 67].)
	IsConnected [▶ 73]	Gets a value indicating whether this instance is connected. (Inherited from Session [▶ 67].)
	Name [▶ 74]	Gets the name of the session (Inherited from Session [▶ 67].)
	NetId [▶ 357]	Gets the NetId of the Session
	Owner [▶ 358]	Gets the Session owner.
	Port [▶ 358]	Gets the Ams Port of the Session

	Name	Description
	Settings [▶ 359]	Gets the settings of the connection.
	Statistics [▶ 359]	Gets the Communication / Session statistics.
	SymbolServer [▶ 74]	Gets the symbol server. (Inherited from Session [▶ 67].)

Methods

	Name	Description
	Close [▶ 76]	Closes this ISession [▶ 60] (Inherited from Session [▶ 67].)
	Connect [▶ 76]	Connects the session. (Inherited from Session [▶ 67].)
	Disconnect [▶ 77]	Disconnects the session from the target. (Inherited from Session [▶ 67].)
	Dispose. [▶ 78]	Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources. (Inherited from Session [▶ 67].)
	Dispose(Boolean) [▶ 361]	Releases unmanaged and - optionally - managed resources. (Overrides Session.Dispose(Boolean) [▶ 79].)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize [▶ 361]	Finalizes an instance of the AdsSession class. (Overrides Object.Finalize..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetSessionName [▶ 362]	Gets the name/string identifier of the session. (Overrides Session.GetSessionName. [▶ 79].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnConnect [▶ 362]	Handler function connecting the Session. (Overrides Session.OnConnect(Boolean) [▶ 80].)
	OnCreateSymbolServer [▶ 363]	Handler function creating the symbol server object. (Overrides Session.OnCreateSymbolServer. [▶ 80].)
	OnDisconnect [▶ 364]	Called when [disconnect]. (Overrides Session.OnDisconnect. [▶ 81].)
	OnGetAddress [▶ 364]	Handler function getting the address of the session. (Overrides Session.OnGetAddress. [▶ 81].)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Events

	Name	Description
	ConnectionStateChanged [▶ 82]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed. (Inherited from Session [▶ 67].)

Fields

	Name	Description
	connection [▶ 84]	The (established) connection (Inherited from Session [▶ 67].)

Remarks

On top of the well known TcAdsClient [▶ 625] class that is used traditionally for ADS communication, the AdsSession class provides the following additionally abilities out of the box: These are used to provide more stable connections to ADS Servers than the TcAdsClient [▶ 625] can provide. The main issues are Resurrection / Self-Healing after communication timeouts, faster and less error prone reaction to communication errors (not necessarily waiting for communication timeouts) und enhanced communication diagnosis. These enhanced features are provided by the following additions to the TwinCAT.Ads API:

- AdsConnection [▶ 148] class.
- Enhanced diagnosis in form of communication statistics Statistics [▶ 359]
- (semi-automatic) Resurrectable client communication with AdsConnection [▶ 148] objects.
- Symbol caching SymbolServer [▶ 74]
- Fail fast handler for connection stabilization IFailFastHandler [▶ 523]

The AdsConnection [▶ 148] is established by calling the Connect. [▶ 76] method. The returned AdsConnection [▶ 148] can be used as long the AdsSession exists.

Examples

The following sample shows a simple use of the AdsSession object. The AdsSession object (and the dynamic SymbolLoader features) are only available from .NET 4 and upwards.

Use of the AdsSession object

```
using System;
using System.Diagnostics;
using System.Threading;

using TwinCAT;
using TwinCAT.Ads;
using TwinCAT.Ads.TypeSystem;
using TwinCAT.TypeSystem;
using TwinCAT.TypeSystem.Generic;

namespace Sample
{
    class Session
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        AmsAddress address = ArgParser.Parse(args);
        SessionSettings settings = SessionSettings.Default; // Default settings are Async access wit
h Timeout 5 sec

        // Async access is necessary for Console applications!

        using (AdsSession session = new AdsSession(address, settings))
        {
        AdsConnection connection = (AdsConnection) session.Connect(); // Establish the connection
        connection.ConnectionStateChanged += Connection_ConnectionStateChanged;

        ConnectionState connectionState = connection.ConnectionState; // The actual connection state

        // Read the identification and version number of the device
        DeviceInfo deviceInfo = connection.ReadDeviceInfo();
        Version version = deviceInfo.Version.ConvertToStandard();
        Console.WriteLine(string.Format("DeviceName: {0}", deviceInfo.Name));
        Console.WriteLine(string.Format("DeviceVersion: {0}", version.ToString(3)));

        /// Read the state of the device
        StateInfo stateInfo = connection.ReadState();
        AdsState adsState = stateInfo.AdsState;

        short deviceState = stateInfo.DeviceState;
        Console.WriteLine(string.Format("DeviceState: {0}", deviceState));
        Console.WriteLine(string.Format("AdsState   : {0}", adsState));
```

```
        // Other ADS methods (as formerly used on TcAdsClient) can be used also on connection object
:

        // connection.Read(...)
        // connection.Write(...)
        // connection.AddDeviceNotificationEx += ...


        // Session communication Diagnostic:

        int resurrectionTries = connection.ResurrectingTries;
        int succeededResurrections = connection.Resurrections;

        AdsCommunicationStatistics statistics = session.Statistics; // The communication statistics

        // Symbol access:
        // The Session holds and Caches the Symbolic data information
        ReadOnlyDataTypeCollection types = session.SymbolServer.DataTypes;
        ReadOnlySymbolCollection symbols = session.SymbolServer.Symbols;

        dynamic projectNameSymbol = symbols["TwinCAT_SystemInfoVarList._AppInfo.ProjectName"];
        string projectName = (string) projectNameSymbol.ReadValue();

        // Or use dynamic objects
        dynamic appInfo = symbols["TwinCAT_SystemInfoVarList._AppInfo"];
        string projectName2 = appInfo.ProjectName.ReadValue();

        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
        }
    }

    private static void Connection_ConnectionStateChanged(object sender, ConnectionStateChangedEvent
Args e)
    {
        Console.WriteLine("Connection State changed (NewState: {0}, OldState: {1}",e.NewState,e.OldS
tate);
    }
```

Reference

TwinCAT.Ads Namespace [▶ 108]

TwinCAT.Session [▶ 67]

TwinCAT.Ads.IAdsSession [▶ 516]

IInterceptionFactory

5.2.20.1 AdsSession Constructor

Overload List

	Name	Description
	AdsSession(AmsAddress) [▶ 352]	Initializes a new instance of the AdsSession [▶ 347] class.
	AdsSession(AmsAddress, SessionSettings) [▶ 352]	Initializes a new instance of the AdsSession [▶ 347] class.
	AdsSession(AmsNetId, Int32) [▶ 353]	Initializes a new instance of the AdsSession [▶ 347] class.

	Name	Description
	AdsSession(AmsAddress, SessionSettings, Object) [▶ 354]	Initializes a new instance of the AdsSession [▶ 347] class.
	AdsSession(AmsNetId, Int32, SessionSettings) [▶ 354]	Initializes a new instance of the AdsSession [▶ 347] class.

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.1.1 AdsSession Constructor (AmsAddress)

Initializes a new instance of the AdsSession [▶ 347] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsSession(
    AmsAddress address
)
```

VB

```
Public Sub New (
    address As AmsAddress
)
```

Parameters

address	Type: TwinCAT.Ads.AmsAddress [▶ 410] The address.

Reference

AdsSession Class [▶ 347]

AdsSession Overload [▶ 351]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.1.2 AdsSession Constructor (AmsAddress, SessionSettings)

Initializes a new instance of the AdsSession [▶ 347] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsSession(
    AmsAddress address,
    SessionSettings settings
)
```

VB

```
Public Sub New (
    address As AmsAddress,
    settings As SessionSettings
)
```

Parameters

address	Type: TwinCAT.Ads.AmsAddress [▶ 410] The address.
settings	Type: TwinCAT.Ads.SessionSettings [▶ 592] The settings.

Reference

AdsSession Class [▶ 347]

AdsSession Overload [▶ 351]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.1.3 AdsSession Constructor (AmsNetId, Int32)

Initializes a new instance of the AdsSession [▶ 347] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsSession(
    AmsNetId netId,
    int port
)
```

VB

```
Public Sub New (
    netId As AmsNetId,
    port As Integer
)
```

Parameters

netId	Type: TwinCAT.Ads.AmsNetId [▶ 428] The net identifier.
port	Type: System.Int32 The port.

Reference

AdsSession Class [▶ 347]

AdsSession Overload [▶ 351]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.1.4 AdsSession Constructor (AmsAddress, SessionSettings, Object)

Initializes a new instance of the AdsSession [▶ 347] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsSession(
    AmsAddress address,
    SessionSettings settings,
    Object owner
)
```

VB

```
Public Sub New (
    address As AmsAddress,
    settings As SessionSettings,
    owner As Object
)
```

Parameters

address	Type: TwinCAT.Ads.AmsAddress [▶ 410] The address.
settings	Type: TwinCAT.Ads.SessionSettings [▶ 592] The settings.
owner	Type: System.Object The session owner

Reference

AdsSession Class [▶ 347]

AdsSession Overload [▶ 351]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.1.5 AdsSession Constructor (AmsNetId, Int32, SessionSettings)

Initializes a new instance of the AdsSession [▶ 347] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsSession(
    AmsNetId netId,
    int port,
    SessionSettings settings
)
```

VB

```
Public Sub New (
    netId As AmsNetId,
    port As Integer,
    settings As SessionSettings
)
```

Parameters

netId	Type: TwinCAT.Ads.AmsNetId [▶ 428] The net identifier.
port	Type: System.Int32 The port.
settings	Type: TwinCAT.Ads.SessionSettings [▶ 592] The settings.

Reference

AdsSession Class [▶ 347]

AdsSession Overload [▶ 351]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.2 AdsSession Properties

The AdsSession [▶ 347] type exposes the following members.

Properties

	Name	Description
	Address [▶ 356]	Gets the target address of the AdsSession [▶ 347]
	AddressSpecifier [▶ 69]	Gets the communication endpoint address string representation. (Inherited from Session [▶ 67].)
	Connection [▶ 357]	Gets the connection.
	ConnectionState [▶ 70]	Gets the current Connection state of the Session [▶ 67] (Inherited from Session [▶ 67].)
	Disposed [▶ 72]	Gets a value indicating whether this Session [▶ 67] is disposed. (Inherited from Session [▶ 67].)
	EstablishedAt [▶ 72]	Gets the UTC time when the session was established. (Inherited from Session [▶ 67].)
	Id [▶ 73]	Gets the Session Identifier (Inherited from Session [▶ 67].)

	Name	Description
	IsConnected [▶ 73]	Gets a value indicating whether this instance is connected. (Inherited from Session [▶ 67].)
	Name [▶ 74]	Gets the name of the session (Inherited from Session [▶ 67].)
	NetId [▶ 357]	Gets the NetId of the Session
	Owner [▶ 358]	Gets the Session owner.
	Port [▶ 358]	Gets the Ams Port of the Session
	Settings [▶ 359]	Gets the settings of the connection.
	Statistics [▶ 359]	Gets the Communication / Session statistics.
	SymbolServer [▶ 74]	Gets the symbol server. (Inherited from Session [▶ 67].)

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.2.1 AdsSession.Address Property

Gets the target address of the AdsSession [▶ 347]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress Address { get; }
```

VB

```
Public ReadOnly Property Address As AmsAddress
    Get
```

Property Value

Type: AmsAddress [▶ 410]
The address.

Implements

IAdsSession.Address [▶ 518]

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.2.2 AdsSession.Connection Property

Gets the connection.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsConnection Connection { get; }
```

VB

```
Public ReadOnly Property Connection As AdsConnection
    Get
```

Property Value

Type: AdsConnection [▶ 148]
The connection.

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.2.3 AdsSession.NetId Property

Gets the NetId of the Session

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsNetId NetId { get; }
```

VB

```
Public ReadOnly Property NetId As AmsNetId
    Get
```

Property Value

Type: AmsNetId [▶ 428]
The net identifier.

Implements

IAdsSession.NetId [▶ 519]

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.2.4 AdsSession.Owner Property

Gets the Session owner.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object Owner { get; }
```

VB

```
Public ReadOnly Property Owner As Object
    Get
```

Property Value

Type: Object
The owner or NULL

Implements

IAdsSession.Owner [▶ 519]

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.2.5 AdsSession.Port Property

Gets the Ams Port of the Session

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Port { get; }
```

VB

```
Public ReadOnly Property Port As Integer
    Get
```

Property Value

Type: Int32
The port.

Implements

IAdsSession.Port [▶ 520]

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.2.6 AdsSession.Settings Property

Gets the settings of the connection.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SessionSettings Settings { get; }
```

VB

```
Public ReadOnly Property Settings As SessionSettings
    Get
```

Property Value

Type: SessionSettings [▶ 592]
The settings.

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.2.7 AdsSession.Statistics Property

Gets the Communication / Session statistics.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsCommunicationStatistics Statistics { get; }
```

VB

```
Public ReadOnly Property Statistics As AdsCommunicationStatistics
    Get
```

Property Value

Type: AdsCommunicationStatistics [▶ 139]
The communication / Session statistics.

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.3 AdsSession Methods

The AdsSession [▶ 347] type exposes the following members.

Methods

	Name	Description
	Close [▶ 76]	Closes this ISession [▶ 60] (Inherited from Session [▶ 67].)
	Connect [▶ 76]	Connects the session. (Inherited from Session [▶ 67].)
	Disconnect [▶ 77]	Disconnects the session from the target. (Inherited from Session [▶ 67].)
	Dispose. [▶ 78]	Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources. (Inherited from Session [▶ 67].)
	Dispose(Boolean) [▶ 361]	Releases unmanaged and - optionally - managed resources. (Overrides Session.Dispose(Boolean) [▶ 79].)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize [▶ 361]	Finalizes an instance of the AdsSession [▶ 347] class. (Overrides Object.Finalize..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetSessionName [▶ 362]	Gets the name/string identifier of the session. (Overrides Session.GetSessionName. [▶ 79].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnConnect [▶ 362]	Handler function connecting the Session. (Overrides Session.OnConnect(Boolean) [▶ 80].)
	OnCreateSymbolServer [▶ 363]	Handler function creating the symbol server object. (Overrides Session.OnCreateSymbolServer. [▶ 80].)
	OnDisconnect [▶ 364]	Called when [disconnect]. (Overrides Session.OnDisconnect. [▶ 81].)
	OnGetAddress [▶ 364]	Handler function getting the address of the session. (Overrides Session.OnGetAddress. [▶ 81].)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.3.1 AdsSession.Dispose Method

Overload List

	Name	Description
	Dispose. [▶ 78]	Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources. (Inherited from Session [▶ 67].)
	Dispose(Boolean) [▶ 361]	Releases unmanaged and - optionally - managed resources. (Overrides Session.Dispose(Boolean) [▶ 79].)

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

AdsSession.Dispose Method (Boolean)

Releases unmanaged and - optionally - managed resources.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected override void Dispose(
    bool disposing
)
```

VB

```
Protected Overrides Sub Dispose (
    disposing As Boolean
)
```

Parameters

disposing	Type: System.Boolean true to release both managed and unmanaged resources; false to release only unmanaged resources.

Reference

AdsSession Class [▶ 347]

Dispose Overload [▶ 361]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.3.2 AdsSession.Finalize Method

Finalizes an instance of the AdsSession [▶ 347] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected override void Finalize()
```

VB

```
Protected Overrides Sub Finalize
```

Implements

Object.Finalize.

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.3.3 AdsSession.GetSessionName Method

Gets the name/string identifier of the session.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected override string GetSessionName()
```

VB

```
Protected Overrides Function GetSessionName As String
```

Return Value

Type: String
System.String.

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.3.4 AdsSession.OnConnect Method

Handler function connecting the Session.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected override IConnection OnConnect(
    bool reconnect
)
```

VB

```
Protected Overrides Function OnConnect (
    reconnect As Boolean
) As IConnection
```

Parameters

reconnect	Type: System.Boolean

Return Value

Type: IConnection [▶ 46]
IConnection.

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.3.5 AdsSession.OnCreateSymbolServer Method

Handler function creating the symbol server object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected override ISymbolServer OnCreateSymbolServer()
```

VB

```
Protected Overrides Function OnCreateSymbolServer As ISymbolServer
```

Return Value

Type: ISymbolServer [▶ 1649]
ISymbolServer.

Exceptions

Exception	Condition
SessionNotConnectedException [▶ 97]	The connection is not established!

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.3.6 AdsSession.OnDisconnect Method

Called when [disconnect].

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected override bool OnDisconnect()
```

VB

```
Protected Overrides Function OnDisconnect As Boolean
```

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.3.7 AdsSession.OnGetAddress Method

Handler function getting the address of the session.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected override string OnGetAddress()
```

VB

```
Protected Overrides Function OnGetAddress As String
```

Return Value

Type: String
System.String.

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.4 AdsSession Events

The AdsSession [▶ 347] type exposes the following members.

Events

	Name	Description
	ConnectionStateChanged [▶ 82]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed. (Inherited from Session [▶ 67].)

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.20.5 AdsSession Fields

The AdsSession [▶ 347] type exposes the following members.

Fields

	Name	Description
	connection [▶ 84]	The (established) connection (Inherited from Session [▶ 67].)

Reference

AdsSession Class [▶ 347]

TwinCAT.Ads Namespace [▶ 108]

5.2.21 AdsState Enumeration

Describes the AdsState.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public enum AdsState
```

VB

```
Public Enumeration AdsState
```

Members

	Member name	Value	Description
	Invalid	0	Invalid
	Idle	1	Idle
	Reset	2	Reset
	Init	3	Initialize

	Member name	Value	Description
	Start	4	Start
	Run	5	Run
	Stop	6	Stop
	SaveConfig	7	Save Configuration
	LoadConfig	8	Load Configuration
	PowerFailure	9	Power failure
	PowerGood	10	Power Good
	Error	11	Error
	Shutdown	12	Shutdown
	Suspend	13	Suspend
	Resume	14	Resume
	Config	15	Config (System is in config mode)
	Reconfig	16	Reconfig (System should restart in config mode)
	Stopping	17	Stopping
	Incompatible	18	Incompatible
	Exception	19	Exception

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.22 AdsStateChangedEventArgs Class

Provides data for AdsStateChangedEvent of the class TcAdsClient.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.Ads.AdsStateChangedEventArgs

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class AdsStateChangedEventArgs : EventArgs
```

VB

```
Public NotInheritable Class AdsStateChangedEventArgs
    Inherits EventArgs
```

The AdsStateChangedEventArgs type exposes the following members.

Constructors

	Name	Description
	AdsStateChangedEventArgs(AdsStateChangedEventArgs) [▶ 367]	Initializes a new instance of the AdsStateChangedEventArgs class.

	Name	Description
	AdsStateChangedEventArgs(StateInfo) [▶ 368]	Initializes a new instance of the AdsStateChangedEventArgs class.

Properties

	Name	Description
	State [▶ 369]	Current state of the ADS device.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.22.1 AdsStateChangedEventArgs Constructor

Overload List

	Name	Description
	AdsStateChangedEventArgs(AdsStateChangedEventArgs) [▶ 367]	Initializes a new instance of the AdsStateChangedEventArgs class.
	AdsStateChangedEventArgs(StateInfo) [▶ 368]	Initializes a new instance of the AdsStateChangedEventArgs class.

Reference

AdsStateChangedEventArgs Class [▶ 366]

TwinCAT.Ads Namespace [▶ 108]

5.2.22.1.1 AdsStateChangedEventArgs Constructor (AdsStateChangedEventArgs)

Initializes a new instance of the AdsStateChangedEventArgs class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsStateChangedEventArgs(
    AdsStateChangedEventArgs eventArgs
)
```

VB

```
Public Sub New (
    eventArgs As AdsStateChangedEventArgs
)
```

Parameters

eventArgs	Type: TwinCAT.Ads.AdsStateChangedEventArgs [▶ 366] The AdsStateChangedEventArgs [▶ 366] instance containing the event data.

Reference

AdsStateChangedEventArgs Class [▶ 366]

AdsStateChangedEventArgs Overload [▶ 367]

TwinCAT.Ads Namespace [▶ 108]

5.2.22.1.2 AdsStateChangedEventArgs Constructor (StateInfo)

Initializes a new instance of the AdsStateChangedEventArgs class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsStateChangedEventArgs(
    StateInfo state
)
```

VB

```
Public Sub New (
    state As StateInfo
)
```

Parameters

state	Type: TwinCAT.Ads.StateInfo [▶ 598] Current state of the ADS device.

Reference

AdsStateChangedEventArgs Class [▶ 366]

AdsStateChangedEventArgs Overload [▶ 367]

TwinCAT.Ads Namespace [▶ 108]

5.2.22.2 AdsStateChangedEventArgs Properties

The AdsStateChangedEventArgs [▶ 366] type exposes the following members.

Properties

	Name	Description
	State [▶ 369]	Current state of the ADS device.

Reference

AdsStateChangedEventArgs Class [▶ 366]

TwinCAT.Ads Namespace [▶ 108]

5.2.22.2.1 AdsStateChangedEventArgs.State Property

Current state of the ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public StateInfo State { get; }
```

VB

```
Public ReadOnly Property State As StateInfo
    Get
```

Property Value

Type: StateInfo [▶ 598]
The state.

Reference

AdsStateChangedEventArgs Class [▶ 366]

TwinCAT.Ads Namespace [▶ 108]

5.2.22.3 AdsStateChangedEventArgs Methods

The AdsStateChangedEventArgs [▶ 366] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)

	Name	Description
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

AdsStateChangedEventArgs Class [▶ 366]

TwinCAT.Ads Namespace [▶ 108]

5.2.23 AdsStateChangedEventArgs2 Class

Event Arguments for AdsStateChanged events.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.Ads.AdsStateChangedEventArgs2
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class AdsStateChangedEventArgs2 : EventArgs
```

VB

```
Public NotInheritable Class AdsStateChangedEventArgs2
    Inherits EventArgs
```

The AdsStateChangedEventArgs2 type exposes the following members.

Constructors

	Name	Description
	AdsStateChangedEventArgs2 [▶ 371]	Initializes a new instance of the AdsStateChangedEventArgs2 class.

Properties

	Name	Description
	Connection [▶ 372]	Gets the connection.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Fields

	Name	Description
	NewState [▶ 373]	The new state
	OldState [▶ 373]	The old state
	Session [▶ 374]	The session

Reference

TwinCAT.Ads Namespace [▶ 108]

System.EventArgs

5.2.23.1 AdsStateChangedEventArgs2 Constructor

Initializes a new instance of the AdsStateChangedEventArgs2 [▶ 370] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsStateChangedEventArgs2(
    StateInfo newState,
    StateInfo oldState,
    IAdsSession session
)
```

VB

```
Public Sub New (
    newState As StateInfo,
    oldState As StateInfo,
    session As IAdsSession
)
```

Parameters

newState	Type: TwinCAT.Ads.StateInfo [▶ 598] The new state.
oldState	Type: TwinCAT.Ads.StateInfo [▶ 598] The old state.
session	Type: TwinCAT.Ads.IAdsSession [▶ 516] The session.

Reference

AdsStateChangedEventArgs2 Class [▶ 370]

TwinCAT.Ads Namespace [▶ 108]

5.2.23.2 AdsStateChangedEventArgs2 Properties

The AdsStateChangedEventArgs2 [▶ 370] type exposes the following members.

Properties

	Name	Description
	Connection [▶ 372]	Gets the connection.

Reference

AdsStateChangedEventArgs2 Class [▶ 370]

TwinCAT.Ads Namespace [▶ 108]

5.2.23.2.1 AdsStateChangedEventArgs2.Connection Property

Gets the connection.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IAdsConnection Connection { get; }
```

VB

```
Public ReadOnly Property Connection As IAdsConnection
    Get
```

Property Value

Type: IAdsConnection [▶ 469]
The connection.

Reference

AdsStateChangedEventArgs2 Class [▶ 370]

TwinCAT.Ads Namespace [▶ 108]

5.2.23.3 AdsStateChangedEventArgs2 Methods

The AdsStateChangedEventArgs2 [▶ 370] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

AdsStateChangedEventArgs2 Class [▶ 370]

TwinCAT.Ads Namespace [▶ 108]

5.2.23.4 AdsStateChangedEventArgs2 Fields

The AdsStateChangedEventArgs2 [▶ 370] type exposes the following members.

Fields

	Name	Description
	NewState [▶ 373]	The new state
	OldState [▶ 373]	The old state
	Session [▶ 374]	The session

Reference

AdsStateChangedEventArgs2 Class [▶ 370]

TwinCAT.Ads Namespace [▶ 108]

5.2.23.4.1 AdsStateChangedEventArgs2.NewState Field

The new state

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly StateInfo NewState
```

VB

```
Public ReadOnly NewState As StateInfo
```

Field Value

Type: StateInfo [▶ 598]

Reference

AdsStateChangedEventArgs2 Class [▶ 370]

TwinCAT.Ads Namespace [▶ 108]

5.2.23.4.2 AdsStateChangedEventArgs2.OldState Field

The old state

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly StateInfo OldState
```

VB

```
Public ReadOnly OldState As StateInfo
```

Field Value

Type: StateInfo [▶ 598]

Reference

AdsStateChangedEventArgs2 Class [▶ 370]

TwinCAT.Ads Namespace [▶ 108]

5.2.23.4.3 AdsStateChangedEventArgs2.Session Field

The session

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly IAdsSession Session
```

VB

```
Public ReadOnly Session As IAdsSession
```

Field Value

Type: IAdsSession [▶ 516]

Reference

AdsStateChangedEventArgs2 Class [▶ 370]

TwinCAT.Ads Namespace [▶ 108]

5.2.24 AdsStateChangedEventHandler Delegate

Event handler for the AdsStateChanged event in the class TcAdsClient.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public delegate void AdsStateChangedEventHandler(
    Object sender,
    AdsStateChangedEventArgs e
)
```

VB

```
Public Delegate Sub AdsStateChangedEventHandler (
    sender As Object,
    e As AdsStateChangedEventArgs
)
```

Parameters

sender	Type: System.Object
e	Type: TwinCAT.Ads.AdsStateChangedEventArgs [▶ 366]

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.25 AdsStream Class

The class AdsStream is a stream class used for ADS communication.

Inheritance Hierarchy

System.Object
  System.MarshalByRefObject
    System.IO.Stream
      System.IO.MemoryStream
        TwinCAT.Ads.AdsStream

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class AdsStream : MemoryStream
```

VB

```
Public Class AdsStream
    Inherits MemoryStream
```

The AdsStream type exposes the following members.

Constructors

	Name	Description
	AdsStream. [▶ 379]	Initializes a new instance of the AdsStream class. The instance has an expandable capacity initialized to zero.
	AdsStream(.Byte.) [▶ 379]	Initializes a new instance of the AdsStream class, based on a byte array.
	AdsStream(Int32) [▶ 380]	Initializes a new instance of the AdsStream class.
	AdsStream(.Byte., Int32, Int32) [▶ 380]	Initializes a new instance of the AdsStream class, based on the specified region of a byte array.

Properties

	Name	Description
	CanRead	Gets a value indicating whether the current stream supports reading. (Inherited from MemoryStream.)
	CanSeek	Gets a value indicating whether the current stream supports seeking. (Inherited from MemoryStream.)
	CanTimeout	Gets a value that determines whether the current stream can time out. (Inherited from Stream.)
	CanWrite	Gets a value indicating whether the current stream supports writing. (Inherited from MemoryStream.)
	Capacity	Gets or sets the number of bytes allocated for this stream. (Inherited from MemoryStream.)
	Length	Gets the length of the stream in bytes. (Inherited from MemoryStream.)
	Position	Gets or sets the current position within the stream. (Inherited from MemoryStream.)
	ReadTimeout	Gets or sets a value, in miliseconds, that determines how long the stream will attempt to read before timing out. (Inherited from Stream.)
	WriteTimeout	Gets or sets a value, in miliseconds, that determines how long the stream will attempt to write before timing out. (Inherited from Stream.)

Methods

	Name	Description
	BeginRead	Begins an asynchronous read operation. (Consider using ReadAsync(.Byte., Int32, Int32) instead; see the Remarks section.) (Inherited from Stream.)
	BeginWrite	Begins an asynchronous write operation. (Consider using WriteAsync(.Byte., Int32, Int32) instead; see the Remarks section.) (Inherited from Stream.)
	Close	Closes the current stream and releases any resources (such as sockets and file handles) associated with the current stream. Instead of calling this method, ensure that the stream is properly disposed. (Inherited from Stream.)
	CopyTo(Stream)	Reads the bytes from the current stream and writes them to another stream. (Inherited from Stream.)
	CopyTo(Stream, Int32)	Reads the bytes from the current stream and writes them to another stream, using a specified buffer size. (Inherited from Stream.)
	CopyToAsync(Stream)	Asynchronously reads the bytes from the current stream and writes them to another stream. (Inherited from Stream.)
	CopyToAsync(Stream, Int32)	Asynchronously reads the bytes from the current stream and writes them to another stream, using a specified buffer size. (Inherited from Stream.)
	CopyToAsync(Stream, Int32, CancellationToken)	Asynchronously reads all the bytes from the current stream and writes them to another stream, using a specified buffer size and cancellation token. (Inherited from MemoryStream.)
	CreateObjRef	Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
	CreateWaitHandle	**Obsolete.** Allocates a WaitHandle object. (Inherited from Stream.)

	Name	Description
	Dispose.	Releases all resources used by the Stream. (Inherited from Stream.)
	Dispose(Boolean)	Releases the unmanaged resources used by the MemoryStream class and optionally releases the managed resources. (Inherited from MemoryStream.)
	EndRead	Waits for the pending asynchronous read to complete. (Consider using ReadAsync(.Byte., Int32, Int32) instead; see the Remarks section.) (Inherited from Stream.)
	EndWrite	Ends an asynchronous write operation. (Consider using WriteAsync(.Byte., Int32, Int32) instead; see the Remarks section.) (Inherited from Stream.)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	Flush	Overrides the Flush. method so that no action is performed. (Inherited from MemoryStream.)
	FlushAsync.	Asynchronously clears all buffers for this stream and causes any buffered data to be written to the underlying device. (Inherited from Stream.)
	FlushAsync(CancellationToken)	Asynchronously clears all buffers for this stream, and monitors cancellation requests. (Inherited from MemoryStream.)
	GetBuffer	Returns the array of unsigned bytes from which this stream was created. (Inherited from MemoryStream.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetLifetimeService	Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	InitializeLifetimeService	Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
	MemberwiseClone.	Creates a shallow copy of the current Object. (Inherited from Object.)
	MemberwiseClone(Boolean)	Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
	ObjectInvariant	**Obsolete.** Provides support for a Contract. (Inherited from Stream.)
	Read	Reads a block of bytes from the current stream and writes the data to a buffer. (Inherited from MemoryStream.)
	ReadAsync(.Byte., Int32, Int32)	Asynchronously reads a sequence of bytes from the current stream and advances the position within the stream by the number of bytes read. (Inherited from Stream.)
	ReadAsync(.Byte., Int32, Int32, CancellationToken)	Asynchronously reads a sequence of bytes from the current stream, advances the position within the stream by the number of bytes read, and monitors cancellation requests. (Inherited from MemoryStream.)
	ReadByte	Reads a byte from the current stream. (Inherited from MemoryStream.)
	Seek	Sets the position within the current stream to the specified value. (Inherited from MemoryStream.)

	Name	Description
	SetLength	Sets the length of the current stream to the specified value. (Inherited from MemoryStream.)
	ToArray	Writes the stream contents to a byte array, regardless of the Position property. (Inherited from MemoryStream.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetBuffer	Returns the array of unsigned bytes from which this stream was created. The return value indicates whether the conversion succeeded. (Inherited from MemoryStream.)
	Write	Writes a block of bytes to the current stream using data read from a buffer. (Inherited from MemoryStream.)
	WriteAsync(.Byte., Int32, Int32)	Asynchronously writes a sequence of bytes to the current stream and advances the current position within this stream by the number of bytes written. (Inherited from Stream.)
	WriteAsync(.Byte., Int32, Int32, CancellationToken)	Asynchronously writes a sequence of bytes to the current stream, advances the current position within this stream by the number of bytes written, and monitors cancellation requests. (Inherited from MemoryStream.)
	WriteByte	Writes a byte to the current stream at the current position. (Inherited from MemoryStream.)
	WriteTo	Writes the entire contents of this memory stream to another stream. (Inherited from MemoryStream.)

Fields

	Name	Description
	origin [▶ 384]	The origin of the stream.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.25.1 AdsStream Constructor

Overload List

	Name	Description
	AdsStream. [▶ 379]	Initializes a new instance of the AdsStream class. The instance has an expandable capacity initialized to zero.
	AdsStream(.Byte.) [▶ 379]	Initializes a new instance of the AdsStream class, based on a byte array.
	AdsStream(Int32) [▶ 380]	Initializes a new instance of the AdsStream class.
	AdsStream(.Byte., Int32, Int32) [▶ 380]	Initializes a new instance of the AdsStream class, based on the specified region of a byte array.

Reference

AdsStream Class [▶ 375]

TwinCAT.Ads Namespace [▶ 108]

5.2.25.1.1 AdsStream Constructor

Initializes a new instance of the AdsStream class. The instance has an expandable capacity initialized to zero.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsStream()
```

VB

```
Public Sub New
```

Reference

AdsStream Class [▶ 375]

AdsStream Overload [▶ 378]

TwinCAT.Ads Namespace [▶ 108]

5.2.25.1.2 AdsStream Constructor (.Byte.)

Initializes a new instance of the AdsStream class, based on a byte array.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsStream(
    byte[] buffer
)
```

VB

```
Public Sub New (
    buffer As Byte()
)
```

Parameters

buffer	Type: .System.Byte. The array of bytes from which to create this stream.

Reference

AdsStream Class [▶ 375]

AdsStream Overload [▶ 378]

TwinCAT.Ads Namespace [▶ 108]

5.2.25.1.3 AdsStream Constructor (Int32)

Initializes a new instance of the AdsStream class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsStream(
    int length
)
```

VB

```
Public Sub New (
    length As Integer
)
```

Parameters

length	Type: System.Int32 Length of the stream in bytes.

Reference

AdsStream Class [▶ 375]

AdsStream Overload [▶ 378]

TwinCAT.Ads Namespace [▶ 108]

5.2.25.1.4 AdsStream Constructor (.Byte., Int32, Int32)

Initializes a new instance of the AdsStream class, based on the specified region of a byte array.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsStream(
    byte[] buffer,
    int offset,
    int length
)
```

VB

```
Public Sub New (
    buffer As Byte(),
    offset As Integer,
    length As Integer
)
```

Parameters

buffer	Type: .System.Byte. The array of bytes from which to create this stream.

offset	Type: System.Int32 The offset in buffer at which the stream begins.
length	Type: System.Int32 The length of the stream in bytes.

Reference

AdsStream Class [▶ 375]

AdsStream Overload [▶ 378]

TwinCAT.Ads Namespace [▶ 108]

5.2.25.2 AdsStream Properties

The AdsStream [▶ 375] type exposes the following members.

Properties

	Name	Description
	CanRead	Gets a value indicating whether the current stream supports reading. (Inherited from MemoryStream.)
	CanSeek	Gets a value indicating whether the current stream supports seeking. (Inherited from MemoryStream.)
	CanTimeout	Gets a value that determines whether the current stream can time out. (Inherited from Stream.)
	CanWrite	Gets a value indicating whether the current stream supports writing. (Inherited from MemoryStream.)
	Capacity	Gets or sets the number of bytes allocated for this stream. (Inherited from MemoryStream.)
	Length	Gets the length of the stream in bytes. (Inherited from MemoryStream.)
	Position	Gets or sets the current position within the stream. (Inherited from MemoryStream.)
	ReadTimeout	Gets or sets a value, in miliseconds, that determines how long the stream will attempt to read before timing out. (Inherited from Stream.)
	WriteTimeout	Gets or sets a value, in miliseconds, that determines how long the stream will attempt to write before timing out. (Inherited from Stream.)

Reference

AdsStream Class [▶ 375]

TwinCAT.Ads Namespace [▶ 108]

5.2.25.3 AdsStream Methods

The AdsStream [▶ 375] type exposes the following members.

Methods

	Name	Description
	BeginRead	Begins an asynchronous read operation. (Consider using ReadAsync(.Byte., Int32, Int32) instead; see the Remarks section.) (Inherited from Stream.)

	Name	Description
	BeginWrite	Begins an asynchronous write operation. (Consider using WriteAsync(.Byte., Int32, Int32) instead; see the Remarks section.) (Inherited from Stream.)
	Close	Closes the current stream and releases any resources (such as sockets and file handles) associated with the current stream. Instead of calling this method, ensure that the stream is properly disposed. (Inherited from Stream.)
	CopyTo(Stream)	Reads the bytes from the current stream and writes them to another stream. (Inherited from Stream.)
	CopyTo(Stream, Int32)	Reads the bytes from the current stream and writes them to another stream, using a specified buffer size. (Inherited from Stream.)
	CopyToAsync(Stream)	Asynchronously reads the bytes from the current stream and writes them to another stream. (Inherited from Stream.)
	CopyToAsync(Stream, Int32)	Asynchronously reads the bytes from the current stream and writes them to another stream, using a specified buffer size. (Inherited from Stream.)
	CopyToAsync(Stream, Int32, CancellationToken)	Asynchronously reads all the bytes from the current stream and writes them to another stream, using a specified buffer size and cancellation token. (Inherited from MemoryStream.)
	CreateObjRef	Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
	CreateWaitHandle	**Obsolete.** Allocates a WaitHandle object. (Inherited from Stream.)
	Dispose.	Releases all resources used by the Stream. (Inherited from Stream.)
	Dispose(Boolean)	Releases the unmanaged resources used by the MemoryStream class and optionally releases the managed resources. (Inherited from MemoryStream.)
	EndRead	Waits for the pending asynchronous read to complete. (Consider using ReadAsync(.Byte., Int32, Int32) instead; see the Remarks section.) (Inherited from Stream.)
	EndWrite	Ends an asynchronous write operation. (Consider using WriteAsync(.Byte., Int32, Int32) instead; see the Remarks section.) (Inherited from Stream.)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	Flush	Overrides the Flush. method so that no action is performed. (Inherited from MemoryStream.)
	FlushAsync.	Asynchronously clears all buffers for this stream and causes any buffered data to be written to the underlying device. (Inherited from Stream.)
	FlushAsync(CancellationToken)	Asynchronously clears all buffers for this stream, and monitors cancellation requests. (Inherited from MemoryStream.)
	GetBuffer	Returns the array of unsigned bytes from which this stream was created. (Inherited from MemoryStream.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)

	Name	Description
	GetLifetimeService	Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	InitializeLifetimeService	Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
	MemberwiseClone.	Creates a shallow copy of the current Object. (Inherited from Object.)
	MemberwiseClone(Boolean)	Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
	ObjectInvariant	**Obsolete.** Provides support for a Contract. (Inherited from Stream.)
	Read	Reads a block of bytes from the current stream and writes the data to a buffer. (Inherited from MemoryStream.)
	ReadAsync(.Byte., Int32, Int32)	Asynchronously reads a sequence of bytes from the current stream and advances the position within the stream by the number of bytes read. (Inherited from Stream.)
	ReadAsync(.Byte., Int32, Int32, CancellationToken)	Asynchronously reads a sequence of bytes from the current stream, advances the position within the stream by the number of bytes read, and monitors cancellation requests. (Inherited from MemoryStream.)
	ReadByte	Reads a byte from the current stream. (Inherited from MemoryStream.)
	Seek	Sets the position within the current stream to the specified value. (Inherited from MemoryStream.)
	SetLength	Sets the length of the current stream to the specified value. (Inherited from MemoryStream.)
	ToArray	Writes the stream contents to a byte array, regardless of the Position property. (Inherited from MemoryStream.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetBuffer	Returns the array of unsigned bytes from which this stream was created. The return value indicates whether the conversion succeeded. (Inherited from MemoryStream.)
	Write	Writes a block of bytes to the current stream using data read from a buffer. (Inherited from MemoryStream.)
	WriteAsync(.Byte., Int32, Int32)	Asynchronously writes a sequence of bytes to the current stream and advances the current position within this stream by the number of bytes written. (Inherited from Stream.)
	WriteAsync(.Byte., Int32, Int32, CancellationToken)	Asynchronously writes a sequence of bytes to the current stream, advances the current position within this stream by the number of bytes written, and monitors cancellation requests. (Inherited from MemoryStream.)
	WriteByte	Writes a byte to the current stream at the current position. (Inherited from MemoryStream.)
	WriteTo	Writes the entire contents of this memory stream to another stream. (Inherited from MemoryStream.)

Reference

AdsStream Class [▶ 375]

TwinCAT.Ads Namespace [▶ 108]

5.2.25.4 AdsStream Fields

The AdsStream [▶ 375] type exposes the following members.

Fields

	Name	Description
	origin [▶ 384]	The origin of the stream.

Reference

AdsStream Class [▶ 375]

TwinCAT.Ads Namespace [▶ 108]

5.2.25.4.1 AdsStream.origin Field

The origin of the stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected int origin
```

VB

```
Protected origin As Integer
```

Field Value

Type: Int32

Reference

AdsStream Class [▶ 375]

TwinCAT.Ads Namespace [▶ 108]

5.2.26 AdsSumCommandException Class

The exception that is thrown when an ADS SumCommand error occurs.

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.Ads.AdsErrorException [▶ 310]
          TwinCAT.Ads.AdsSumCommandException

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class AdsSumCommandException : AdsErrorException
```

VB

```
<SerializableAttribute>
Public Class AdsSumCommandException
    Inherits AdsErrorException
```

The AdsSumCommandException type exposes the following members.

Constructors

	Name	Description
	AdsSumCommandException [▶ 386]	Initializes a new Instance of the AdsErrorException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	ErrorCode [▶ 314]	Gets the error code of the Exception. (Inherited from AdsErrorException [▶ 310].)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	SumCommand [▶ 387]	Gets the sum command.
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)

	Name	Description
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData [▶ 388]	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Overrides AdsErrorException.GetObjectData(SerializationInfo, StreamingContext) [▶ 317].)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.26.1 AdsSumCommandException Constructor

Initializes a new Instance of the AdsErrorException class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsSumCommandException(
    string message,
    ISumCommand command
)
```

VB

```
Public Sub New (
    message As String,
    command As ISumCommand
)
```

Parameters

message	Type: System.String The message.
command	Type: TwinCAT.Ads.SumCommand.ISumCommand [▶ 893] The command.

Reference

AdsSumCommandException Class [▶ 384]

TwinCAT.Ads Namespace [▶ 108]

5.2.26.2 AdsSumCommandException Properties

The AdsSumCommandException [▶ 384] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	ErrorCode [▶ 314]	Gets the error code of the Exception. (Inherited from AdsErrorException [▶ 310].)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	SumCommand [▶ 387]	Gets the sum command.
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

AdsSumCommandException Class [▶ 384]

TwinCAT.Ads Namespace [▶ 108]

5.2.26.2.1 AdsSumCommandException.SumCommand Property

Gets the sum command.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ISumCommand SumCommand { get; }
```

VB

```
Public ReadOnly Property SumCommand As ISumCommand
    Get
```

Property Value

Type: ISumCommand [▶ 893]
The sum command.

Reference

AdsSumCommandException Class [▶ 384]

TwinCAT.Ads Namespace [▶ 108]

5.2.26.3 AdsSumCommandException Methods

The AdsSumCommandException [▶ 384] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData [▶ 388]	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Overrides AdsErrorException.GetObjectData(SerializationInfo, StreamingContext) [▶ 317].)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

AdsSumCommandException Class [▶ 384]

TwinCAT.Ads Namespace [▶ 108]

5.2.26.3.1 AdsSumCommandException.GetObjectData Method

When overridden in a derived class, sets the SerializationInfo with information about the exception.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override void GetObjectData(
    SerializationInfo info,
    StreamingContext context
)
```

VB

```
Public Overrides Sub GetObjectData (
    info As SerializationInfo,
    context As StreamingContext
)
```

Parameters

info	Type: System.Runtime.Serialization.SerializationInfo The SerializationInfo that holds the serialized object data about the exception being thrown.
context	Type: System.Runtime.Serialization.StreamingContext The StreamingContext that contains contextual information about the source or destination.

Implements

ISerializable.GetObjectData(SerializationInfo, StreamingContext)
_Exception.GetObjectData(SerializationInfo, StreamingContext)

Exceptions

Exception	Condition
ArgumentNullException	info

Reference

AdsSumCommandException Class [▶ 384]

TwinCAT.Ads Namespace [▶ 108]

5.2.26.4 AdsSumCommandException Events

The AdsSumCommandException [▶ 384] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

AdsSumCommandException Class [▶ 384]

TwinCAT.Ads Namespace [▶ 108]

5.2.27 AdsSymbolException Class

Symbol Exception

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.Ads.AdsSymbolException
          TwinCAT.Ads.RpcMethodNotSupportedException [▶ 586]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class AdsSymbolException : AdsException
```

VB

```
<SerializableAttribute>
Public Class AdsSymbolException
    Inherits AdsException
```

The AdsSymbolException type exposes the following members.

Constructors

	Name	Description
	AdsSymbolException(String, String) [▶ 392]	Initializes a new instance of the AdsSymbolException class.
	AdsSymbolException(String, ITcAdsSymbol) [▶ 393]	Initializes a new instance of the AdsSymbolException class.
	AdsSymbolException(String, String, Exception) [▶ 393]	Initializes a new instance of the AdsSymbolException class.
	AdsSymbolException(String, ITcAdsSymbol, Exception) [▶ 394]	Initializes a new instance of the AdsSymbolException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)

	Name	Description
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Fields

	Name	Description
	Symbol [▶ 397]	The symbol
	SymbolName [▶ 397]	The symbol

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.27.1 AdsSymbolException Constructor

Overload List

	Name	Description
	AdsSymbolException(String, String) [▶ 392]	Initializes a new instance of the AdsSymbolException [▶ 389] class.
	AdsSymbolException(String, ITcAdsSymbol) [▶ 393]	Initializes a new instance of the AdsSymbolException [▶ 389] class.
	AdsSymbolException(String, String, Exception) [▶ 393]	Initializes a new instance of the AdsSymbolException [▶ 389] class.
	AdsSymbolException(String, ITcAdsSymbol, Exception) [▶ 394]	Initializes a new instance of the AdsSymbolException [▶ 389] class.

Reference

AdsSymbolException Class [▶ 389]

TwinCAT.Ads Namespace [▶ 108]

5.2.27.1.1 AdsSymbolException Constructor (String, String)

Initializes a new instance of the AdsSymbolException [▶ 389] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsSymbolException(
    string message,
    string symbolName
)
```

VB

```
Public Sub New (
    message As String,
    symbolName As String
)
```

Parameters

message	Type: System.String The message.

symbolName	Type: System.String Symbol path.

Reference

AdsSymbolException Class [▶ 389]

AdsSymbolException Overload [▶ 392]

TwinCAT.Ads Namespace [▶ 108]

5.2.27.1.2 AdsSymbolException Constructor (String, ITcAdsSymbol)

Initializes a new instance of the AdsSymbolException [▶ 389] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsSymbolException(
    string message,
    ITcAdsSymbol symbol
)
```

VB

```
Public Sub New (
    message As String,
    symbol As ITcAdsSymbol
)
```

Parameters

message	Type: System.String The message.
symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.

Reference

AdsSymbolException Class [▶ 389]

AdsSymbolException Overload [▶ 392]

TwinCAT.Ads Namespace [▶ 108]

5.2.27.1.3 AdsSymbolException Constructor (String, String, Exception)

Initializes a new instance of the AdsSymbolException [▶ 389] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsSymbolException(
    string message,
    string symbolName,
    Exception innerException
)
```

VB

```
Public Sub New (
    message As String,
    symbolName As String,
    innerException As Exception
)
```

Parameters

message	Type: System.String The message.
symbolName	Type: System.String Symbol path.
innerException	Type: System.Exception The inner exception.

Reference

AdsSymbolException Class [▶ 389]

AdsSymbolException Overload [▶ 392]

TwinCAT.Ads Namespace [▶ 108]

5.2.27.1.4 AdsSymbolException Constructor (String, ITcAdsSymbol, Exception)

Initializes a new instance of the AdsSymbolException [▶ 389] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsSymbolException(
    string message,
    ITcAdsSymbol symbol,
    Exception innerException
)
```

VB

```
Public Sub New (
    message As String,
    symbol As ITcAdsSymbol,
    innerException As Exception
)
```

Parameters

message	Type: System.String The message.

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
innerException	Type: System.Exception The inner exception.

Reference

AdsSymbolException Class [▶ 389]

AdsSymbolException Overload [▶ 392]

TwinCAT.Ads Namespace [▶ 108]

5.2.27.2 AdsSymbolException Properties

The AdsSymbolException [▶ 389] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

AdsSymbolException Class [▶ 389]

TwinCAT.Ads Namespace [▶ 108]

5.2.27.3 AdsSymbolException Methods

The AdsSymbolException [▶ 389] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)

	Name	Description
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

AdsSymbolException Class [▶ 389]

TwinCAT.Ads Namespace [▶ 108]

5.2.27.4 AdsSymbolException Events

The AdsSymbolException [▶ 389] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

AdsSymbolException Class [▶ 389]

TwinCAT.Ads Namespace [▶ 108]

5.2.27.5 AdsSymbolException Fields

The AdsSymbolException [▶ 389] type exposes the following members.

Fields

	Name	Description
	Symbol [▶ 397]	The symbol
	SymbolName [▶ 397]	The symbol

Reference

AdsSymbolException Class [▶ 389]

TwinCAT.Ads Namespace [▶ 108]

5.2.27.5.1 AdsSymbolException.Symbol Field

The symbol

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[NonSerializedAttribute]
public readonly ITcAdsSymbol Symbol
```

VB

```
<NonSerializedAttribute>
Public ReadOnly Symbol As ITcAdsSymbol
```

Field Value

Type: ITcAdsSymbol [▶ 549]

Reference

AdsSymbolException Class [▶ 389]

TwinCAT.Ads Namespace [▶ 108]

5.2.27.5.2 AdsSymbolException.SymbolName Field

The symbol

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[NonSerializedAttribute]
public readonly string SymbolName
```

VB

```
<NonSerializedAttribute>
Public ReadOnly SymbolName As String
```

Field Value

Type: String

Reference

AdsSymbolException Class [▶ 389]

TwinCAT.Ads Namespace [▶ 108]

5.2.28 AdsSymbolVersionChangedEventArgs Class

Provides data for AdsSymbolVersionChangedEvent of the class TcAdsClient.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.Ads.AdsSymbolVersionChangedEventArgs

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class AdsSymbolVersionChangedEventArgs : EventArgs
```

VB

```
Public NotInheritable Class AdsSymbolVersionChangedEventArgs
    Inherits EventArgs
```

The AdsSymbolVersionChangedEventArgs type exposes the following members.

Constructors

	Name	Description
	AdsSymbolVersionChangedEventArgs(Byte) [▶ 399]	Initializes a new instance of the AdsSymbolVersionChangedEventArgs class.
	AdsSymbolVersionChangedEventArgs(AdsSymbolVersionChangedEventArgs) [▶ 400]	Initializes a new instance of the AdsSymbolVersionChangedEventArgs class.

Properties

	Name	Description
	SymbolVersion [▶ 400]	Current symbol version device.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.28.1 AdsSymbolVersionChangedEventArgs Constructor

Overload List

	Name	Description
	AdsSymbolVersionChangedEventArgs(Byte) [▶ 399]	Initializes a new instance of the AdsSymbolVersionChangedEventArgs class.
	AdsSymbolVersionChangedEventArgs(AdsSymbolVersionChangedEventArgs) [▶ 400]	Initializes a new instance of the AdsSymbolVersionChangedEventArgs class.

Reference

AdsSymbolVersionChangedEventArgs Class [▶ 398]

TwinCAT.Ads Namespace [▶ 108]

5.2.28.1.1 AdsSymbolVersionChangedEventArgs Constructor (Byte)

Initializes a new instance of the AdsSymbolVersionChangedEventArgs class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsSymbolVersionChangedEventArgs(
    byte symbolVersion
)
```

VB

```
Public Sub New (
    symbolVersion As Byte
)
```

Parameters

symbolVersion	Type: System.Byte Current symbol version.

Reference

AdsSymbolVersionChangedEventArgs Class [▶ 398]

AdsSymbolVersionChangedEventArgs Overload [▶ 399]

TwinCAT.Ads Namespace [▶ 108]

5.2.28.1.2 AdsSymbolVersionChangedEventArgs Constructor (AdsSymbolVersionChangedEventArgs)

Initializes a new instance of the AdsSymbolVersionChangedEventArgs class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsSymbolVersionChangedEventArgs(
    AdsSymbolVersionChangedEventArgs eventArgs
)
```

VB

```
Public Sub New (
    eventArgs As AdsSymbolVersionChangedEventArgs
)
```

Parameters

eventArgs	Type: TwinCAT.Ads.AdsSymbolVersionChangedEventArgs [▶ 398] The AdsSymbolVersionChangedEventArgs [▶ 398] instance containing the event data.

Reference

AdsSymbolVersionChangedEventArgs Class [▶ 398]

AdsSymbolVersionChangedEventArgs Overload [▶ 399]

TwinCAT.Ads Namespace [▶ 108]

5.2.28.2 AdsSymbolVersionChangedEventArgs Properties

The AdsSymbolVersionChangedEventArgs [▶ 398] type exposes the following members.

Properties

	Name	Description
	SymbolVersion [▶ 400]	Current symbol version device.

Reference

AdsSymbolVersionChangedEventArgs Class [▶ 398]

TwinCAT.Ads Namespace [▶ 108]

5.2.28.2.1 AdsSymbolVersionChangedEventArgs.SymbolVersion Property

Current symbol version device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public short SymbolVersion { get; }
```

VB

```
Public ReadOnly Property SymbolVersion As Short
    Get
```

Property Value

Type: Int16
The symbol version.

Reference

AdsSymbolVersionChangedEventArgs Class [▶ 398]

TwinCAT.Ads Namespace [▶ 108]

5.2.28.3 AdsSymbolVersionChangedEventArgs Methods

The AdsSymbolVersionChangedEventArgs [▶ 398] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

AdsSymbolVersionChangedEventArgs Class [▶ 398]

TwinCAT.Ads Namespace [▶ 108]

5.2.29 AdsTransMode Enumeration

ADS Transmission Mode for ADS Notifications.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public enum AdsTransMode
```

VB

```
Public Enumeration AdsTransMode
```

Members

	Member name	Value	Description
	None	0	None / Uninitalized transport mode. No AdsNotification [▶ 774] event is fired.
	ClientCycle	1	Client triggered cyclic AdsNotification [▶ 774] event. The AdsNotification [▶ 774] event is fired cyclically triggered from the client side. Polling is used from the User Application to read values, before they are fired as Notifications. Client side triggering has the following consequences: • The realtime environment on the server side will be less stressed (expecially the mailbox queue). • Value requests are serialized one after another and are handled slower (synchronouly, not asynchronously) • Implicit synchronization of the events into the UI Thread.
	ClientOnChange	2	The AdsNotification [▶ 774] event is fired when data changes triggered by the client. The AdsNotification [▶ 774] event is fired on-change triggered from the client side. Polling is used from the User Application to read values, before they are fired as Notifications. Client side triggering has the following consequences: • The realtime environment on the server side will be less stressed (expecially the mailbox queue). • Value requests are serialized one after another and are handled slower (synchronouly, not asynchronously) • Implicit synchronization of the events into the UI Thread.
	Cyclic	3	The AdsNotification [▶ 774] event is fired cyclically. The Notification will be registered on the ADS Server side for a cyclical trigger (dependant on time parameter) and is bound to the 'default' task of the addressed target. In case of the PLC target (e.g. Port 851) the default task is the first configured task. Each time the 'default' task has finished its cycle the realtime system will check for the expired cycle time and sends the AdsNotification [▶ 774] message on expiry.

	Member name	Value	Description
			The used ContextMask for the 'default' task is 0. **Please be aware, that server side 'Change' notifications stress the realtime system and should be handled with care. Therefore, dependent of the cycle time of the task and the capabilities of the system only a limited set of Cyclic Notifications should be used!** A system limit for server side notification registrations is 1024.
	OnChange	4	On-Change AdsNotification [▶ 774] event. The Notification will be registered on the ADS Server side for an on-change and optional cyclical trigger (dependant on parameters) and is bound to the 'default' task of the addressed target. In case of the PLC target (e.g. Port 851) the default task is the first configured task. Each time this task has finished its cycle the realtime system will check for the changed value and an optional expired cycle time and sends the AdsNotification [▶ 774] message on change or expiry. The used ContextMask for the 'default' task is 0. **Please be aware, that server side 'OnChange' notifications stress the realtime system / the default task with value comparisons. Therefore, dependent of the cycle time of the task and the capabilities of the system a higher amount of notification registrations should be handled with care !** A system limit for server side notification registrations is 1024.
	CyclicInContext	5	The AdsNotification [▶ 774] event is fired cyclically within the given task context. A Value of parameter is interpreted as task context number ContextMask [▶ 555]. This can be important, if the notifications have to be synchron with specific tasks, but should not be used in the default case. The Notification will be registered on the ADS Server side for a cyclical trigger (dependant on time parameter) and is bound to the task specified by the ContextMask of the addressed target. In case of the PLC

	Member name	Value	Description
			target (e.g. Port 851) the ContextMask is the Index of the global TASKINFOARRAY - 1. Each time this task has finished its cycle the realtime system will check for the expired cycle time and sends the AdsNotification [▶ 774] message on expiry.
	OnChangeInContext	6	The AdsNotification [▶ 774] event is fired when the data changes within the given task context. A Value of parameter is interpreted as task context number ContextMask [▶ 555]. This can be important, if the notifications have to be synchron with specific tasks, but should not be used in the default case. The Notification will be registered on the ADS Server side for an on-change and optional cyclical trigger (dependant on parameters) and is bound to the task specified by the ContextMask of the addressed target. In case of the PLC target (e.g. Port 851) the ContextMask is the Index of the global TASKINFOARRAY - 1. Each time this task has finished its cycle the realtime system will check for the changed value and an optional expired cycle time and sends the AdsNotification [▶ 774] message on change or expiry. **Please be aware, that server side 'OnChange' notifications stress the realtime system / the default task with value comparisons. Therefore, dependent of the cycle time of the task and the capabilities of the system only a limited set of OnChange Notifications should be used!**

Remarks

The AdsTransMode configures the registration of the AdsNotification [▶ 774] at the server system and how the parameters of the AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 665] are interpreted. The following general scenarios are addressed:

- Cyclic notifications.
- Notifications on value change.
- Server side and Client side notifications.
- Binding of notifications to specific tasks.

In the default case the OnChange or the Cyclic (Server cycle) should be used. All other modes are side cases for special purposes.

More about the AdsNotifications: ADS Notification concept [▶ 20].

Reference

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

AddDeviceNotification Overload [▶ 657]

AddDeviceNotificationEx Overload [▶ 673]

5.2.30 AdsVersion Structure

The structure contains the version number, revision number and build number.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public struct AdsVersion
```

VB

```
Public Structure AdsVersion
```

The AdsVersion type exposes the following members.

Constructors

	Name	Description
	AdsVersion [▶ 406]	Initializes a new instance of the AdsVersion struct.

Properties

	Name	Description
	Build [▶ 407]	Gets or sets the build number.
	IsEmpty [▶ 407]	Gets a value indicating whether this instance is empty / uninitialized.
	Revision [▶ 408]	Gets or sets the revision number.
	Version [▶ 408]	Gets or sets the version number.

Methods

	Name	Description
	ConvertToStandard [▶ 409]	Converts this AdsVersion to a .NET Framework Version [▶ 408] object.
	Equals	Indicates whether this instance and a specified object are equal. (Inherited from ValueType.)
	GetHashCode	Returns the hash code for this instance. (Inherited from ValueType.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)

	Name	Description
	ToString	Returns the fully qualified type name of this instance. (Inherited from ValueType.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.30.1 AdsVersion Constructor

Initializes a new instance of the AdsVersion [▶ 405] struct.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsVersion(
    int version,
    int revision,
    int build
)
```

VB

```
Public Sub New (
    version As Integer,
    revision As Integer,
    build As Integer
)
```

Parameters

version	Type: System.Int32 The version.
revision	Type: System.Int32 The revision.
build	Type: System.Int32 The build.

Exceptions

Exception	Condition
ArgumentOutOfRangeException	version or revision

Reference

AdsVersion Structure [▶ 405]

TwinCAT.Ads Namespace [▶ 108]

5.2.30.2 AdsVersion Properties

The AdsVersion [▶ 405] type exposes the following members.

Properties

	Name	Description
	Build [▶ 407]	Gets or sets the build number.
	IsEmpty [▶ 407]	Gets a value indicating whether this instance is empty / uninitialized.
	Revision [▶ 408]	Gets or sets the revision number.
	Version [▶ 408]	Gets or sets the version number.

Reference

AdsVersion Structure [▶ 405]

TwinCAT.Ads Namespace [▶ 108]

5.2.30.2.1 AdsVersion.Build Property

Gets or sets the build number.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Build { get; set; }
```

VB

```
Public Property Build As Integer
    Get
    Set
```

Property Value

Type: Int32

Reference

AdsVersion Structure [▶ 405]

TwinCAT.Ads Namespace [▶ 108]

5.2.30.2.2 AdsVersion.IsEmpty Property

Gets a value indicating whether this instance is empty / uninitialized.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsEmpty { get; }
```

VB

```
Public ReadOnly Property IsEmpty As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is empty; otherwise, false.

Reference

AdsVersion Structure [▶ 405]

TwinCAT.Ads Namespace [▶ 108]

5.2.30.2.3 AdsVersion.Revision Property

Gets or sets the revision number.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public byte Revision { get; set; }
```

VB

```
Public Property Revision As Byte
    Get
    Set
```

Property Value

Type: Byte

Reference

AdsVersion Structure [▶ 405]

TwinCAT.Ads Namespace [▶ 108]

5.2.30.2.4 AdsVersion.Version Property

Gets or sets the version number.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public byte Version { get; set; }
```

VB

```
Public Property Version As Byte
    Get
    Set
```

Property Value

Type: Byte

Reference

AdsVersion Structure [▶ 405]

TwinCAT.Ads Namespace [▶ 108]

5.2.30.3 AdsVersion Methods

The AdsVersion [▶ 405] type exposes the following members.

Methods

	Name	Description
	ConvertToStandard [▶ 409]	Converts this AdsVersion [▶ 405] to a .NET Framework Version [▶ 408] object.
	Equals	Indicates whether this instance and a specified object are equal. (Inherited from ValueType.)
	GetHashCode	Returns the hash code for this instance. (Inherited from ValueType.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns the fully qualified type name of this instance. (Inherited from ValueType.)

Reference

AdsVersion Structure [▶ 405]

TwinCAT.Ads Namespace [▶ 108]

5.2.30.3.1 AdsVersion.ConvertToStandard Method

Converts this AdsVersion [▶ 405] to a .NET Framework Version [▶ 408] object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Version ConvertToStandard()
```

VB

```
Public Function ConvertToStandard As Version
```

Return Value

Type: Version
Version.

Reference

AdsVersion Structure [▶ 405]

TwinCAT.Ads Namespace [▶ 108]

5.2.31 AmsAddress Class

Ams/Ads Address

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.AmsAddress
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class AmsAddress
```

VB

```
Public Class AmsAddress
```

The AmsAddress type exposes the following members.

Constructors

	Name	Description
	AmsAddress. [▶ 412]	Protected constructor
	AmsAddress(Int32) [▶ 413]	Constructor
	AmsAddress(String) [▶ 413]	Initializes a new instance of the AmsAddress class.
	AmsAddress(AmsAddress) [▶ 414]	Copy constructor
	AmsAddress(AmsPort) [▶ 415]	Constructor
	AmsAddress(.Byte., Int32) [▶ 415]	Constructor
	AmsAddress(.Byte., AmsPort) [▶ 416]	Constructor
	AmsAddress(String, Int32) [▶ 416]	Constructor
	AmsAddress(String, AmsPort) [▶ 417]	Constructor
	AmsAddress(AmsNetId, Int32) [▶ 418]	Constructor

	Name	Description
	AmsAddress(AmsNetId, AmsPort) [▶ 418]	Constructor

Properties

	Name	Description
	NetId [▶ 419]	Gets the NetId
	Port [▶ 420]	Gets the Port number

Methods

	Name	Description
	Clone [▶ 421]	Clones this instance.
	Equals [▶ 421]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 422]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
S	Parse [▶ 422]	Parses a string to an AmsAddress object.
	ToString [▶ 423]	Converts the Address to String 'NetId:Port' (Overrides Object.ToString..)
S	TryParse [▶ 423]	Tries to parse the AmsAddress from string.

Operators

	Name	Description
S	Equality [▶ 425]	Operator==
S	Inequality [▶ 425]	Implements the != operator.

Fields

	Name	Description
	netId [▶ 426]	The net id
	port [▶ 427]	The port

	Name	Description
	RegularExpressionPattern [▶ 427]	The regular expression pattern for AmsAddress

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.31.1 AmsAddress Constructor

Overload List

	Name	Description
	AmsAddress. [▶ 412]	Protected constructor
	AmsAddress(Int32) [▶ 413]	Constructor
	AmsAddress(String) [▶ 413]	Initializes a new instance of the AmsAddress [▶ 410] class.
	AmsAddress(AmsAddress) [▶ 414]	Copy constructor
	AmsAddress(AmsPort) [▶ 415]	Constructor
	AmsAddress(.Byte., Int32) [▶ 415]	Constructor
	AmsAddress(.Byte., AmsPort) [▶ 416]	Constructor
	AmsAddress(String, Int32) [▶ 416]	Constructor
	AmsAddress(String, AmsPort) [▶ 417]	Constructor
	AmsAddress(AmsNetId, Int32) [▶ 418]	Constructor
	AmsAddress(AmsNetId, AmsPort) [▶ 418]	Constructor

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.1.1 AmsAddress Constructor

Protected constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected AmsAddress()
```

VB

```
Protected Sub New
```

Reference

AmsAddress Class [▶ 410]

AmsAddress Overload [▶ 412]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.1.2 AmsAddress Constructor (Int32)

Constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress(
    int port
)
```

VB

```
Public Sub New (
    port As Integer
)
```

Parameters

port	Type: System.Int32 The port.

Reference

AmsAddress Class [▶ 410]

AmsAddress Overload [▶ 412]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.1.3 AmsAddress Constructor (String)

Initializes a new instance of the AmsAddress [▶ 410] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress(
    string str
)
```

VB

```
Public Sub New (
    str As String
)
```

Parameters

str	Type: System.String The address coded as string (Format NetId:Port, 1.2.3.4.5.6:Port)

Reference

AmsAddress Class [▶ 410]

AmsAddress Overload [▶ 412]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.1.4 AmsAddress Constructor (AmsAddress)

Copy constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress(
    AmsAddress address
)
```

VB

```
Public Sub New (
    address As AmsAddress
)
```

Parameters

address	Type: TwinCAT.Ads.AmsAddress [▶ 410] The address.

Reference

AmsAddress Class [▶ 410]

AmsAddress Overload [▶ 412]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.1.5 AmsAddress Constructor (AmsPort)

Constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress(
    AmsPort port
)
```

VB

```
Public Sub New (
    port As AmsPort
)
```

Parameters

port	Type: TwinCAT.Ads.AmsPort [▶ 450] The port.

Reference

AmsAddress Class [▶ 410]

AmsAddress Overload [▶ 412]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.1.6 AmsAddress Constructor (.Byte., Int32)

Constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress(
    byte[] netId,
    int port
)
```

VB

```
Public Sub New (
    netId As Byte(),
    port As Integer
)
```

Parameters

netId	Type: .System.Byte. The net identifier.
port	Type: System.Int32 The port.

Reference

AmsAddress Class [▶ 410]

AmsAddress Overload [▶ 412]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.1.7 AmsAddress Constructor (.Byte., AmsPort)

Constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress(
    byte[] netId,
    AmsPort port
)
```

VB

```
Public Sub New (
    netId As Byte(),
    port As AmsPort
)
```

Parameters

netId	Type: .System.Byte. The net identifier.
port	Type: TwinCAT.Ads.AmsPort [▶ 450] The port.

Reference

AmsAddress Class [▶ 410]

AmsAddress Overload [▶ 412]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.1.8 AmsAddress Constructor (String, Int32)

Constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress(
    string netId,
    int port
)
```

VB

```
Public Sub New (
    netId As String,
    port As Integer
)
```

Parameters

netId	Type: System.String The net identifier.
port	Type: System.Int32 The port.

Reference

AmsAddress Class [▶ 410]

AmsAddress Overload [▶ 412]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.1.9 AmsAddress Constructor (String, AmsPort)

Constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress(
    string netId,
    AmsPort port
)
```

VB

```
Public Sub New (
    netId As String,
    port As AmsPort
)
```

Parameters

netId	Type: System.String The net identifier.
port	Type: TwinCAT.Ads.AmsPort [▶ 450] The port.

Reference

AmsAddress Class [▶ 410]

AmsAddress Overload [▶ 412]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.1.10 AmsAddress Constructor (AmsNetId, Int32)

Constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress(
    AmsNetId netId,
    int port
)
```

VB

```
Public Sub New (
    netId As AmsNetId,
    port As Integer
)
```

Parameters

netId	Type: TwinCAT.Ads.AmsNetId [▶ 428] Net Id
port	Type: System.Int32 Port

Reference

AmsAddress Class [▶ 410]

AmsAddress Overload [▶ 412]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.1.11 AmsAddress Constructor (AmsNetId, AmsPort)

Constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress(
    AmsNetId netId,
    AmsPort port
)
```

VB

```
Public Sub New (
    netId As AmsNetId,
    port As AmsPort
)
```

Parameters

netId	Type: TwinCAT.Ads.AmsNetId [▶ 428] The net identifier.
port	Type: TwinCAT.Ads.AmsPort [▶ 450] The port.

Reference

AmsAddress Class [▶ 410]

AmsAddress Overload [▶ 412]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.2 AmsAddress Properties

The AmsAddress [▶ 410] type exposes the following members.

Properties

	Name	Description
	NetId [▶ 419]	Gets the NetId
	Port [▶ 420]	Gets the Port number

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.2.1 AmsAddress.NetId Property

Gets the NetId

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsNetId NetId { get; set; }
```

VB

```
Public Property NetId As AmsNetId
    Get
    Set
```

Property Value

Type: AmsNetId [▶ 428]
The net identifier.

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.2.2 AmsAddress.Port Property

Gets the Port number

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Port { get; set; }
```

VB

```
Public Property Port As Integer
    Get
    Set
```

Property Value

Type: Int32
The port.

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.3 AmsAddress Methods

The AmsAddress [▶ 410] type exposes the following members.

Methods

	Name	Description
	Clone [▶ 421]	Clones this instance.
	Equals [▶ 421]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 422]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
S	Parse [▶ 422]	Parses a string to an AmsAddress [▶ 410] object.

	Name	Description
	ToString [▶ 423]	Converts the Address to String 'NetId:Port' (Overrides Object.ToString..)
 	TryParse [▶ 423]	Tries to parse the AmsAddress [▶ 410] from string.

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.3.1 AmsAddress.Clone Method

Clones this instance.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress Clone()
```

VB

```
Public Function Clone As AmsAddress
```

Return Value

Type: AmsAddress [▶ 410]
AmsAddress.

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.3.2 AmsAddress.Equals Method

Equals

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool Equals(
    Object obj
)
```

VB

```
Public Overrides Function Equals (
    obj As Object
) As Boolean
```

Parameters

obj	Type: System.Object The object to compare with the current object.

Return Value

Type: Boolean
true if the specified Object is equal to this instance; otherwise, false.

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.3.3 AmsAddress.GetHashCode Method

Gets the HashCode of the Address

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override int GetHashCode()
```

VB

```
Public Overrides Function GetHashCode As Integer
```

Return Value

Type: Int32
A hash code for this instance, suitable for use in hashing algorithms and data structures like a hash table.

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.3.4 AmsAddress.Parse Method

Parses a string to an AmsAddress [▶ 410] object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static AmsAddress Parse(
    string str
)
```

VB

```
Public Shared Function Parse (
    str As String
) As AmsAddress
```

Parameters

str	Type: System.String The string.

Return Value

Type: AmsAddress [▶ 410]
AmsAddress.

Exceptions

Exception	Condition
FormatException	

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.3.5 AmsAddress.ToString Method

Converts the Address to String 'NetId:Port'

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.3.6 AmsAddress.TryParse Method

Tries to parse the AmsAddress [▶ 410] from string.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool TryParse(
    string str,
    out AmsAddress address
)
```

VB

```
Public Shared Function TryParse (
    str As String,
    <OutAttribute> ByRef address As AmsAddress
) As Boolean
```

Parameters

str	Type: System.String The STR.
address	Type: TwinCAT.Ads.AmsAddress [▶ 410]. The address.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.4 AmsAddress Operators

The AmsAddress [▶ 410] type exposes the following members.

Operators

	Name	Description
	Equality [▶ 425]	Operator==
	Inequality [▶ 425]	Implements the != operator.

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.4.1 AmsAddress.Equality Operator

Operator==

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator ==(
    AmsAddress o1,
    AmsAddress o2
)
```

VB

```
Public Shared Operator = (
    o1 As AmsAddress,
    o2 As AmsAddress
) As Boolean
```

Parameters

o1	Type: TwinCAT.Ads.AmsAddress [▶ 410] The o1.
o2	Type: TwinCAT.Ads.AmsAddress [▶ 410] The o2.

Return Value

Type: Boolean
The result of the operator.

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.4.2 AmsAddress.Inequality Operator

Implements the != operator.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator !=(
    AmsAddress o1,
    AmsAddress o2
)
```

VB

```
Public Shared Operator <> (
    o1 As AmsAddress,
    o2 As AmsAddress
) As Boolean
```

Parameters

o1	Type: TwinCAT.Ads.AmsAddress [▶ 410] The o1.
o2	Type: TwinCAT.Ads.AmsAddress [▶ 410] The o2.

Return Value

Type: Boolean
The result of the operator.

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.5 AmsAddress Fields

The AmsAddress [▶ 410] type exposes the following members.

Fields

	Name	Description
	netId [▶ 426]	The net id
	port [▶ 427]	The port
	RegularExpressionPattern [▶ 427]	The regular expression pattern for AmsAddress [▶ 410]

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.5.1 AmsAddress.netId Field

The net id

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected AmsNetId netId
```

VB

```
Protected netId As AmsNetId
```

Field Value

Type: AmsNetId [▶ 428]

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.5.2 AmsAddress.port Field

The port

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected int port
```

VB

```
Protected port As Integer
```

Field Value

Type: Int32

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.31.5.3 AmsAddress.RegularExpressionPattern Field

The regular expression pattern for AmsAddress [▶ 410]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public const string RegularExpressionPattern = "^(?<AmsNetId>((?<First>\d{1,3})\.(?<Second>\d{1,3})
\.(?<Third>\d{1,3})\.(?<Fourth>\d{1,3})\.(?<Fifth>\d{1,3})\.(?<Sixth>\d{1,3})) | Local | Empty |
 LocalHost)(:(?<AdsPort>\d+))?$"
```

VB

```
Public Const RegularExpressionPattern As String = "^(?<AmsNetId>((?<First>\d{1,3})\.(?
<Second>\d{1,3})\.(?<Third>\d{1,3})\.(?<Fourth>\d{1,3})\.(?<Fifth>\d{1,3})\.(?<Sixth>\d{1,3})) |
 Local | Empty | LocalHost)(:(?<AdsPort>\d+))?$"
```

Field Value

Type: String

Reference

AmsAddress Class [▶ 410]

TwinCAT.Ads Namespace [▶ 108]

5.2.32 AmsNetId Class

AMS/ADS Net ID

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.AmsNetId
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class AmsNetId : IComparable<AmsNetId>,
    IComparable
```

VB

```
<SerializableAttribute>
Public Class AmsNetId
    Implements IComparable(Of AmsNetId), IComparable
```

The AmsNetId type exposes the following members.

Constructors

	Name	Description
	AmsNetId(.Byte.) [▶ 430]	Constructor
	AmsNetId(String) [▶ 431]	Constructor
	AmsNetId(AmsNetId) [▶ 431]	Copy Constructor

Properties

	Name	Description
S	Empty [▶ 432]	Creates an empty NetId ("0.0.0.0.0.0")
	IsLocal [▶ 433]	Is the Address Local?
S	Local [▶ 433]	Gets the Local Net ID (System service must be running)
	LocalHost [▶ 434]	Creates the local NetId ("127.0.0.1.1.1")

	Name	Description
S		

Methods

	Name	Description
	Clone [▶ 436]	Clones the NetId
	CompareTo(Object) [▶ 436]	Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.
	CompareTo(AmsNetId) [▶ 437]	Compares the current object with another object of the same type.
	Equals [▶ 438]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
S	FromBinHexString [▶ 439]	Creates the AmsNetId from bin hex string.
	GetHashCode [▶ 439]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
S	IsEqual [▶ 440]	Determines whether the specified AmsNetIds are equal.
S	IsSameTarget [▶ 440]	Determines whether the AmsNetIds refer to the same target.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	NetIdsEqual(.Byte.) [▶ 441]	Compares the netIds
S	NetIdsEqual(.Byte., .Byte.) [▶ 442]	Compares the NetIds
S	Parse [▶ 443]	Converts the string representation of the address to AmsNetId.
	ToBinHex. [▶ 444]	Converts the AmsNetId to a BinHex string.
S	ToBinHex(AmsNetId) [▶ 444]	Converts the specified AmsNetId to a BinHex string.
	ToBytes [▶ 445]	Converts the NetId object to byte array
	ToString. [▶ 446]	Converts the netId to string (Overrides Object.ToString..)
	ToString(String, IFormatProvider) [▶ 446]	Returns a String that represents this instance.
	TryParse [▶ 447]	Converts the string representation of the address to AmsNetId.

	Name	Description
S		

Operators

	Name	Description
S	Equality [▶ 448]	Operator==
S	Inequality [▶ 449]	Implements the != operator.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.32.1 AmsNetId Constructor

Overload List

	Name	Description
	AmsNetId(.Byte.) [▶ 430]	Constructor
	AmsNetId(String) [▶ 431]	Constructor
	AmsNetId(AmsNetId) [▶ 431]	Copy Constructor

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.1.1 AmsNetId Constructor (.Byte.)

Constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsNetId(
    byte[] netId
)
```

VB

```
Public Sub New (
    netId As Byte()
)
```

Parameters

netId	Type: .System.Byte. Net ID in bytes

Exceptions

Exception	Condition
ArgumentException	Not a valid NetId;netId

Reference

AmsNetId Class [▶ 428]

AmsNetId Overload [▶ 430]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.1.2 AmsNetId Constructor (String)

Constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsNetId(
    string netId
)
```

VB

```
Public Sub New (
    netId As String
)
```

Parameters

netId	Type: System.String NetID as string

Reference

AmsNetId Class [▶ 428]

AmsNetId Overload [▶ 430]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.1.3 AmsNetId Constructor (AmsNetId)

Copy Constructor

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsNetId(
    AmsNetId netId
)
```

VB

```
Public Sub New (
    netId As AmsNetId
)
```

Parameters

netId	Type: TwinCAT.Ads.AmsNetId [▶ 428] Net Id.

Reference

AmsNetId Class [▶ 428]

AmsNetId Overload [▶ 430]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.2 AmsNetId Properties

The AmsNetId [▶ 428] type exposes the following members.

Properties

	Name	Description
	Empty [▶ 432]	Creates an empty NetId ("0.0.0.0.0.0")
	IsLocal [▶ 433]	Is the Address Local?
	Local [▶ 433]	Gets the Local Net ID (System service must be running)
	LocalHost [▶ 434]	Creates the local NetId ("127.0.0.1.1.1")

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.2.1 AmsNetId.Empty Property

Creates an empty NetId ("0.0.0.0.0.0")

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static AmsNetId Empty { get; }
```

VB

```
Public Shared ReadOnly Property Empty As AmsNetId
    Get
```

Property Value

Type: AmsNetId [▶ 428]
The empty.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.2.2 AmsNetId.IsLocal Property

Is the Address Local?

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsLocal { get; }
```

VB

```
Public ReadOnly Property IsLocal As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is local; otherwise, false.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.2.3 AmsNetId.Local Property

Gets the Local Net ID (System service must be running)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static AmsNetId Local { get; }
```

VB

```
Public Shared ReadOnly Property Local As AmsNetId
    Get
```

Property Value

Type: AmsNetId [▶ 428]
The local.

Remarks

The system service must be running

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.2.4 AmsNetId.LocalHost Property

Creates the local NetId ("127.0.0.1.1.1")

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static AmsNetId LocalHost { get; }
```

VB

```
Public Shared ReadOnly Property LocalHost As AmsNetId
    Get
```

Property Value

Type: AmsNetId [▶ 428]
The local host.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3 AmsNetId Methods

The AmsNetId [▶ 428] type exposes the following members.

Methods

	Name	Description
	Clone [▶ 436]	Clones the NetId
	CompareTo(Object) [▶ 436]	Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.
	CompareTo(AmsNetId) [▶ 437]	Compares the current object with another object of the same type.
	Equals [▶ 438]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
S	FromBinHexString [▶ 439]	Creates the AmsNetId [▶ 428] from bin hex string.
	GetHashCode [▶ 439]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
S	IsEqual [▶ 440]	Determines whether the specified AmsNetId [▶ 428]s are equal.
S	IsSameTarget [▶ 440]	Determines whether the AmsNetId [▶ 428]s refer to the same target.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	NetIdsEqual(.Byte.) [▶ 441]	Compares the netIds
S	NetIdsEqual(.Byte., .Byte.) [▶ 442]	Compares the NetIds
S	Parse [▶ 443]	Converts the string representation of the address to AmsNetId [▶ 428].
	ToBinHex. [▶ 444]	Converts the AmsNetId [▶ 428] to a BinHex string.
S	ToBinHex(AmsNetId) [▶ 444]	Converts the specified AmsNetId [▶ 428] to a BinHex string.
	ToBytes [▶ 445]	Converts the NetId object to byte array
	ToString. [▶ 446]	Converts the netId to string (Overrides Object.ToString..)
	ToString(String, IFormatProvider) [▶ 446]	Returns a String that represents this instance.
S	TryParse [▶ 447]	Converts the string representation of the address to AmsNetId [▶ 428].

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3.1 AmsNetId.Clone Method

Clones the NetId

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsNetId Clone()
```

VB

```
Public Function Clone As AmsNetId
```

Return Value

Type: AmsNetId [▶ 428]
The cloned AmsNetId [▶ 428]

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3.2 AmsNetId.CompareTo Method

Overload List

	Name	Description
	CompareTo(Object) [▶ 436]	Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.
	CompareTo(AmsNetId) [▶ 437]	Compares the current object with another object of the same type.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

AmsNetId.CompareTo Method (Object)

Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int CompareTo(
    Object obj
)
```

VB

```
Public Function CompareTo (
    obj As Object
) As Integer
```

Parameters

obj	Type: System.Object An object to compare with this instance.

Return Value

Type: Int32
A value that indicates the relative order of the objects being compared. The return value has these meanings: Value Meaning Less than zero This instance precedes obj in the sort order. Zero This instance occurs in the same position in the sort order as obj. Greater than zero This instance follows obj in the sort order.

Implements

IComparable.CompareTo(Object)

Reference

AmsNetId Class [▶ 428]

CompareTo Overload [▶ 436]

TwinCAT.Ads Namespace [▶ 108]

AmsNetId.CompareTo Method (AmsNetId)

Compares the current object with another object of the same type.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int CompareTo(
    AmsNetId other
)
```

VB

```
Public Function CompareTo (
    other As AmsNetId
) As Integer
```

Parameters

other	Type: TwinCAT.Ads.AmsNetId [▶ 428] An object to compare with this object.

Return Value

Type: Int32
A value that indicates the relative order of the objects being compared. The return value has the following meanings: Value Meaning Less than zero This object is less than the other parameter.Zero This object is equal to other. Greater than zero This object is greater than other.

Implements

IComparable.T..CompareTo(T)

Reference

AmsNetId Class [▶ 428]

CompareTo Overload [▶ 436]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3.3 AmsNetId.Equals Method

Equals

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool Equals(
    Object obj
)
```

VB

```
Public Overrides Function Equals (
    obj As Object
) As Boolean
```

Parameters

obj	Type: System.Object The object to compare with the current object.

Return Value

Type: Boolean
true if the specified Object is equal to this instance; otherwise, false.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3.4 AmsNetId.FromBinHexString Method

Creates the AmsNetId [▶ 428] from bin hex string.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static AmsNetId FromBinHexString(
    string str
)
```

VB

```
Public Shared Function FromBinHexString (
    str As String
) As AmsNetId
```

Parameters

str	Type: System.String The BinHex string.

Return Value

Type: AmsNetId [▶ 428]
AmsNetId.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3.5 AmsNetId.GetHashCode Method

Gets the HashCode of the Address

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override int GetHashCode()
```

VB

```
Public Overrides Function GetHashCode As Integer
```

Return Value

Type: Int32
A hash code for this instance, suitable for use in hashing algorithms and data structures like a hash table.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3.6 AmsNetId.IsEqual Method

Determines whether the specified AmsNetId [▶ 428]s are equal.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool IsEqual(
    AmsNetId netIDA,
    AmsNetId netIDB
)
```

VB

```
Public Shared Function IsEqual (
    netIDA As AmsNetId,
    netIDB As AmsNetId
) As Boolean
```

Parameters

netIDA	Type: TwinCAT.Ads.AmsNetId [▶ 428] The net IDA.
netIDB	Type: TwinCAT.Ads.AmsNetId [▶ 428] The net IDB.

Return Value

Type: Boolean
true if the specified net IDA is equal; otherwise, false.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3.7 AmsNetId.IsSameTarget Method

Determines whether the AmsNetId [▶ 428]s refer to the same target.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool IsSameTarget(
    AmsNetId netIDA,
    AmsNetId netIDB
)
```

VB

```
Public Shared Function IsSameTarget (
    netIDA As AmsNetId,
    netIDB As AmsNetId
) As Boolean
```

Parameters

netIDA	Type: TwinCAT.Ads.AmsNetId [▶ 428] NetID of target system A
netIDB	Type: TwinCAT.Ads.AmsNetId [▶ 428] NetID of target system B

Return Value

Type: Boolean
true if the target systems are the same, otherwise false.

Remarks

In comparison to the IsEqual(AmsNetId, AmsNetId) [▶ 440] or Equals(Object) [▶ 438] methods, this Method alse checks against the LocalHost ID, which means that LocalHost [▶ 434] is the same target as Local [▶ 433]

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

AmsNetId.IsEqual(AmsNetId, AmsNetId) [▶ 440]

5.2.32.3.8 AmsNetId.NetIdsEqual Method

Overload List

	Name	Description
	NetIdsEqual(.Byte.) [▶ 441]	Compares the netIds
S	NetIdsEqual(.Byte., .Byte.) [▶ 442]	Compares the NetIds

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

AmsNetId.NetIdsEqual Method (.Byte.)

Compares the netIds

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool NetIdsEqual(
    byte[] netId
)
```

VB

```
Public Function NetIdsEqual (
    netId As Byte()
) As Boolean
```

Parameters

netId	Type: .System.Byte. NetId in bytes.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

AmsNetId Class [▶ 428]

NetIdsEqual Overload [▶ 441]

TwinCAT.Ads Namespace [▶ 108]

AmsNetId.NetIdsEqual Method (.Byte., .Byte.)

Compares the NetIds

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool NetIdsEqual(
    byte[] netId1,
    byte[] netId2
)
```

VB

```
Public Shared Function NetIdsEqual (
    netId1 As Byte(),
    netId2 As Byte()
) As Boolean
```

Parameters

netId1	Type: .System.Byte. NetID1
netId2	Type: .System.Byte. NetId2

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

AmsNetId Class [▶ 428]

NetIdsEqual Overload [▶ 441]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3.9 AmsNetId.Parse Method

Converts the string representation of the address to AmsNetId [▶ 428].

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static AmsNetId Parse(
    string str
)
```

VB

```
Public Shared Function Parse (
    str As String
) As AmsNetId
```

Parameters

str	Type: System.String The string to parse.

Return Value

Type: AmsNetId [▶ 428]
AmsNetId.

Exceptions

Exception	Condition
FormatException	Format of AmsNetId is not valid!

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3.10 AmsNetId.ToBinHex Method

Overload List

	Name	Description
	ToBinHex. [▶ 444]	Converts the AmsNetId [▶ 428] to a BinHex string.
 	ToBinHex(AmsNetId) [▶ 444]	Converts the specified AmsNetId [▶ 428] to a BinHex string.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

AmsNetId.ToBinHex Method

Converts the AmsNetId [▶ 428] to a BinHex string.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ToBinHex()
```

VB

```
Public Function ToBinHex As String
```

Return Value

Type: String
System.String.

Reference

AmsNetId Class [▶ 428]

ToBinHex Overload [▶ 444]

TwinCAT.Ads Namespace [▶ 108]

AmsNetId.ToBinHex Method (AmsNetId)

Converts the specified AmsNetId [▶ 428] to a BinHex string.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static string ToBinHex(
    AmsNetId netId
)
```

VB

```
Public Shared Function ToBinHex (
    netId As AmsNetId
) As String
```

Parameters

netId	Type: TwinCAT.Ads.AmsNetId [▶ 428] NetId to convert

Return Value

Type: String
System.String.

Reference

AmsNetId Class [▶ 428]

ToBinHex Overload [▶ 444]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3.11 AmsNetId.ToBytes Method

Converts the NetId object to byte array

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public byte[] ToBytes()
```

VB

```
Public Function ToBytes As Byte()
```

Return Value

Type: .Byte.
System.Byte[].

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3.12 AmsNetId.ToString Method

Overload List

	Name	Description
	ToString. [▶ 446]	Converts the netId to string (Overrides Object.ToString..)
	ToString(String, IFormatProvider) [▶ 446]	Returns a String that represents this instance.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

AmsNetId.ToString Method

Converts the netId to string

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

AmsNetId Class [▶ 428]

ToString Overload [▶ 446]

TwinCAT.Ads Namespace [▶ 108]

AmsNetId.ToString Method (String, IFormatProvider)

Returns a String that represents this instance.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ToString(
    string format,
    IFormatProvider formatProvider
)
```

VB

```
Public Function ToString (
    format As String,
    formatProvider As IFormatProvider
) As String
```

Parameters

format	Type: System.String The format.
formatProvider	Type: System.IFormatProvider The format provider.

Return Value

Type: String
A String that represents this instance.

Exceptions

Exception	Condition
FormatException	

Remarks

Formatting	Description
g	Standard formatting
x	Formatting as Hexadecimal (small letters)
X	Formatting as Hexadecimal (big letters)

Reference

AmsNetId Class [▶ 428]

ToString Overload [▶ 446]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.3.13 AmsNetId.TryParse Method

Converts the string representation of the address to AmsNetId [▶ 428].

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool TryParse(
    string str,
    out AmsNetId netId
)
```

VB

```
Public Shared Function TryParse (
    str As String,
    <OutAttribute> ByRef netId As AmsNetId
) As Boolean
```

Parameters

str	Type: System.String The string to parse.
netId	Type: TwinCAT.Ads.AmsNetId [▶ 428]. The parsed AmsNetId [▶ 428].

Return Value

Type: Boolean
true if parsed, false otherwise.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.4 AmsNetId Operators

The AmsNetId [▶ 428] type exposes the following members.

Operators

	Name	Description
	Equality [▶ 448]	Operator==
	Inequality [▶ 449]	Implements the != operator.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.4.1 AmsNetId.Equality Operator

Operator==

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator ==(
    AmsNetId o1,
    AmsNetId o2
)
```

VB

```
Public Shared Operator = (
    o1 As AmsNetId,
    o2 As AmsNetId
) As Boolean
```

Parameters

o1	Type: TwinCAT.Ads.AmsNetId [▶ 428] The o1.
o2	Type: TwinCAT.Ads.AmsNetId [▶ 428] The o2.

Return Value

Type: Boolean
The result of the operator.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.32.4.2 AmsNetId.Inequality Operator

Implements the != operator.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator !=(
    AmsNetId o1,
    AmsNetId o2
)
```

VB

```
Public Shared Operator <> (
    o1 As AmsNetId,
    o2 As AmsNetId
) As Boolean
```

Parameters

o1	Type: TwinCAT.Ads.AmsNetId [▶ 428] The o1.
o2	Type: TwinCAT.Ads.AmsNetId [▶ 428] The o2.

Return Value

Type: Boolean
The result of the operator.

Reference

AmsNetId Class [▶ 428]

TwinCAT.Ads Namespace [▶ 108]

5.2.33 AmsPort Enumeration

AmsPorts

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public enum AmsPort
```

VB

```
Public Enumeration AmsPort
```

Members

	Member name	Value	Description
	Router	1	AMS Router (Port 1)
	Debugger	2	AMS Debugger (Port 2)
	R0_TComServer	10	The TCom Server. Dpc or passive level.
	R0_TComServerTask	11	TCom Serve Task. RT context.
	R0_TComServer_PL	12	TCom Serve Task. Passive level.
	R0_TcDebugger	20	TwinCAT Debugger
	R0_TcDebuggerTask	21	TwinCAT Debugger Task
	R0_LicenseServer	30	The License Server (Port 30)
	Logger	100	Logger (Port 100)
	EventLog	110	Event Logger (Port 110)
	DeviceApplication	120	application for coupler (EK), gateway (EL), etc.
	EventLog_UM	130	Event Logger UM
	EventLog_RT	131	Event Logger RT
	EventLogPublisher	132	Event Logger Publisher
	R0_Realtime	200	R0 Realtime (Port 200)
	R0_Trace	290	R0 Trace (Port 290)

	Member name	Value	Description
	R0_IO	300	R0 IO (Port 300)
	R0_NC	500	NC (R0) (Port 500)
	R0_NCSAF	501	R0 Satzausführung (Port 501)
	R0_NCSVB	511	R0 Satzvorbereitung (Port 511)
	R0_NCINSTANCE	520	Preconfigured Nc2-Nc3-Instance
	R0_ISG	550	R0 ISG (Port 550)
	R0_CNC	600	R0 CNC (Port 600)
	R0_LINE	700	R0 Line (Port 700)
	R0_PLC	800	R0 PLC (Port 800)
	Tc2_Plc1	801	Tc2 PLC RuntimeSystem 1 (Port 801)
	Tc2_Plc2	811	Tc2 PLC RuntimeSystem 2 (Port 811)
	Tc2_Plc3	821	Tc2 PLC RuntimeSystem 3 (Port 821)
	Tc2_Plc4	831	Tc2 PLC RuntimeSystem 4 (Port 831)
	R0_RTS	850	R0 RTS (Port 850)
	CamshaftController	900	Camshaft Controller (R0) (Port 900)
	R0_CAMTOOL	950	R0 CAM Tool (Port 950)
	R0_USER	2000	R0 User (Port 2000)
	SystemService	10000	System Service (AMSPORT_R3_SYSSERV, 10000)
	R3_CTRLPROG	10000	(Port 10000)
	R3_SYSCTRL	10001	(Port 10001)
	R3_SYSSAMPLER	10100	Port 10100
	R3_TCPRAWCONN	10200	Port 10200
	R3_TCPIPSERVER	10201	Port 10201
	R3_SYSMANAGER	10300	Port 10300
	R3_SMSSERVER	10400	Port 10400
	R3_MODBUSSERVER	10500	Port 10500
	R3_AMSLOGGER	10502	Port 10502
	R3_S7SERVER	10600	**Obsolete.** Port 10600
	R3_XMLDATASERVER	10600	Port 10600
	R3_AUTOCONFIG	10700	Port 10700
	R3_PLCCONTROL	10800	Port 10800
	R3_FTPCLIENT	10900	Port 10900
	R3_NCCTRL	11000	Port 11000
	R3_NCINTERPRETER	11500	Port 11500
	R3_GSTINTERPRETER	11600	Port 11600
	R3_STRECKECTRL	12000	Port 12000
	R3_CAMCTRL	13000	Port 13000
	R3_SCOPE	14000	Port 14000
	R3_CONDITIONMON	14100	Port 14100
	R3_SINECH1	15000	Port 15000
	R3_CONTROLNET	16000	Port 16000
	R3_OPCSERVER	17000	Port 17000
	R3_OPCCLIENT	17500	Port 17500
	R3_MAILSERVER	18000	Port 18000
	R3_EL60XX	19000	Port 19000

	Member name	Value	Description
	R3_MANAGEMENT	19100	Port 19100
	R3_MIELEHOME	19200	Port 19200
	R3_CPLINK3	19300	Port 19300
	R3_VNSERVICE	19500	Port 19500
	USEDEFAULT	65535	

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.34 AmsRouterNotificationEventArgs Class

Provides data for AmsRouterNotificationEvent of the class TcAdsClient.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.Ads.AmsRouterNotificationEventArgs
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class AmsRouterNotificationEventArgs : EventArgs
```

VB

```
Public NotInheritable Class AmsRouterNotificationEventArgs
    Inherits EventArgs
```

The AmsRouterNotificationEventArgs type exposes the following members.

Constructors

	Name	Description
	AmsRouterNotificationEventArgs [▶ 453]	Initializes a new instance of the AmsRouterNotificationEventArgs class.

Properties

	Name	Description
	State [▶ 453]	Current state of the AMS Router.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)

	Name	Description
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.34.1 AmsRouterNotificationEventArgs Constructor

Initializes a new instance of the AmsRouterNotificationEventArgs class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsRouterNotificationEventArgs(
    AmsRouterState state
)
```

VB

```
Public Sub New (
    state As AmsRouterState
)
```

Parameters

state	Type: TwinCAT.Ads.AmsRouterState [▶ 455] The current state of the Router.

Reference

AmsRouterNotificationEventArgs Class [▶ 452]

TwinCAT.Ads Namespace [▶ 108]

5.2.34.2 AmsRouterNotificationEventArgs Properties

The AmsRouterNotificationEventArgs [▶ 452] type exposes the following members.

Properties

	Name	Description
	State [▶ 453]	Current state of the AMS Router.

Reference

AmsRouterNotificationEventArgs Class [▶ 452]

TwinCAT.Ads Namespace [▶ 108]

5.2.34.2.1 AmsRouterNotificationEventArgs.State Property

Current state of the AMS Router.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsRouterState State { get; }
```

VB

```
Public ReadOnly Property State As AmsRouterState
    Get
```

Property Value

Type: AmsRouterState [▶ 455]

Reference

AmsRouterNotificationEventArgs Class [▶ 452]

TwinCAT.Ads Namespace [▶ 108]

5.2.34.3 AmsRouterNotificationEventArgs Methods

The AmsRouterNotificationEventArgs [▶ 452] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

AmsRouterNotificationEventArgs Class [▶ 452]

TwinCAT.Ads Namespace [▶ 108]

5.2.35 AmsRouterNotificationEventHandler Delegate

Event handler for the AmsRouterNotification event in the class TcAdsClient.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public delegate void AmsRouterNotificationEventHandler(
    Object sender,
    AmsRouterNotificationEventArgs e
)
```

VB

```
Public Delegate Sub AmsRouterNotificationEventHandler (
    sender As Object,
    e As AmsRouterNotificationEventArgs
)
```

Parameters

sender	Type: System.Object
e	Type: TwinCAT.Ads.AmsRouterNotificationEventArgs [▶ 452]

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.36 AmsRouterState Enumeration

State of the AMS Router.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public enum AmsRouterState
```

VB

```
Public Enumeration AmsRouterState
```

Members

	Member name	Value	Description
	Unknown	-1	Unknown Router State
	Stop	0	AMS Router is stopped.
	Start	1	AMS Router is started.
	Removed	2	AMS Router has been removed.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.37 DeviceInfo Structure

The structure contains the name and the version information of the device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public struct DeviceInfo
```

VB

```
Public Structure DeviceInfo
```

The DeviceInfo type exposes the following members.

Properties

	Name	Description
	IsEmpty [▶ 456]	Gets a value indicating whether this instance is empty.
	Name [▶ 457]	Gets or sets the name of the device.
	Version [▶ 457]	Gets or sets the version information.

Methods

	Name	Description
	Equals	Indicates whether this instance and a specified object are equal. (Inherited from ValueType.)
	GetHashCode	Returns the hash code for this instance. (Inherited from ValueType.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns the fully qualified type name of this instance. (Inherited from ValueType.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.37.1 DeviceInfo Properties

The DeviceInfo [▶ 455] type exposes the following members.

Properties

	Name	Description
	IsEmpty [▶ 456]	Gets a value indicating whether this instance is empty.
	Name [▶ 457]	Gets or sets the name of the device.
	Version [▶ 457]	Gets or sets the version information.

Reference

DeviceInfo Structure [▶ 455]

TwinCAT.Ads Namespace [▶ 108]

5.2.37.1.1 DeviceInfo.IsEmpty Property

Gets a value indicating whether this instance is empty.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsEmpty { get; }
```

VB

```
Public ReadOnly Property IsEmpty As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is empty; otherwise, false.

Reference

DeviceInfo Structure [▶ 455]

TwinCAT.Ads Namespace [▶ 108]

5.2.37.1.2 DeviceInfo.Name Property

Gets or sets the name of the device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Name { get; set; }
```

VB

```
Public Property Name As String
    Get
    Set
```

Property Value

Type: String

Reference

DeviceInfo Structure [▶ 455]

TwinCAT.Ads Namespace [▶ 108]

5.2.37.1.3 DeviceInfo.Version Property

Gets or sets the version information.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsVersion Version { get; set; }
```

VB

```
Public Property Version As AdsVersion
    Get
    Set
```

Property Value

Type: AdsVersion [▶ 405]

Reference

DeviceInfo Structure [▶ 455]

TwinCAT.Ads Namespace [▶ 108]

5.2.37.2 DeviceInfo Methods

The DeviceInfo [▶ 455] type exposes the following members.

Methods

	Name	Description
	Equals	Indicates whether this instance and a specified object are equal. (Inherited from ValueType.)
	GetHashCode	Returns the hash code for this instance. (Inherited from ValueType.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns the fully qualified type name of this instance. (Inherited from ValueType.)

Reference

DeviceInfo Structure [▶ 455]

TwinCAT.Ads Namespace [▶ 108]

5.2.38 IAdsAnyAccess Interface

Interface for accessing ADS 'Any' objects.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IAdsAnyAccess
```

VB

```
Public Interface IAdsAnyAccess
```

Methods

	Name	Description
	ReadAny(Int32, Type) [▶ 461]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(Int32, Type, .Int32.) [▶ 461]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type) [▶ 462]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 463]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAnyString(Int32, Int32, Encoding) [▶ 464]	Reads the string.
	ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 465]	Reads the string
	WriteAny(Int32, Object) [▶ 466]	Writes an object synchronously to an ADS device.
	WriteAny(Int32, Object, .Int32.) [▶ 467]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.
	WriteAny(UInt32, UInt32, Object) [▶ 468]	Writes an object synchronously to an ADS device.
	WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 468]	Writes an object synchronously to an ADS device.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.38.1 IAdsAnyAccess Methods

Methods

	Name	Description
	ReadAny(Int32, Type) [▶ 461]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(Int32, Type, .Int32.) [▶ 461]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type) [▶ 462]	Reads data synchronously from an ADS device and writes it to an object.

	Name	Description
	ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 463]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAnyString(Int32, Int32, Encoding) [▶ 464]	Reads the string.
	ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 465]	Reads the string
	WriteAny(Int32, Object) [▶ 466]	Writes an object synchronously to an ADS device.
	WriteAny(Int32, Object, .Int32.) [▶ 467]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.
	WriteAny(UInt32, UInt32, Object) [▶ 468]	Writes an object synchronously to an ADS device.
	WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 468]	Writes an object synchronously to an ADS device.

Reference

IAdsAnyAccess Interface [▶ 458]

TwinCAT.Ads Namespace [▶ 108]

5.2.38.1.1 IAdsAnyAccess.ReadAny Method

Overload List

	Name	Description
	ReadAny(Int32, Type) [▶ 461]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(Int32, Type, .Int32.) [▶ 461]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type) [▶ 462]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 463]	Reads data synchronously from an ADS device and writes it to an object.

Reference

IAdsAnyAccess Interface [▶ 458]

TwinCAT.Ads Namespace [▶ 108]

IAdsAnyAccess.ReadAny Method (Int32, Type)

Reads data synchronously from an ADS device and writes it to an object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object ReadAny(
    int variableHandle,
    Type type
)
```

VB

```
Function ReadAny (
    variableHandle As Integer,
    type As Type
) As Object
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable.
type	Type: System.Type Type of the object to be read.

Return Value

Type: Object
The object the read data is written to.

Reference

IAdsAnyAccess Interface [▶ 458]

ReadAny Overload [▶ 460]

TwinCAT.Ads Namespace [▶ 108]

IAdsAnyAccess.ReadAny Method (Int32, Type, .Int32.)

Reads data synchronously from an ADS device and writes it to an object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object ReadAny(
    int variableHandle,
    Type type,
    int[] args
)
```

VB

```
Function ReadAny (
    variableHandle As Integer,
    type As Type,
    args As Integer()
) As Object
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable.
type	Type: System.Type Type of the object to be read.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Object
The object the read data is written to.

Remarks

If the Type of the object to be read is a string type, the first element of the parameter args specifies the number of characters of the string. If the Type of the object to be read is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Reference

IAdsAnyAccess Interface [▶ 458]

ReadAny Overload [▶ 460]

TwinCAT.Ads Namespace [▶ 108]

IAdsAnyAccess.ReadAny Method (UInt32, UInt32, Type)

Reads data synchronously from an ADS device and writes it to an object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object ReadAny(
    uint indexGroup,
    uint indexOffset,
    Type type
)
```

VB

```
Function ReadAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    type As Type
) As Object
```

Parameters

indexGroup	Type: System.UInt32 Index group of the ADS variable.
indexOffset	Type: System.UInt32 Index offset of the ADS variable.
type	Type: System.Type Type of the object to be read.

Return Value

Type: Object
The object the read data is written to.

Reference

IAdsAnyAccess Interface [▶ 458]

ReadAny Overload [▶ 460]

TwinCAT.Ads Namespace [▶ 108]

IAdsAnyAccess.ReadAny Method (UInt32, UInt32, Type, .Int32.)

Reads data synchronously from an ADS device and writes it to an object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object ReadAny(
    uint indexGroup,
    uint indexOffset,
    Type type,
    int[] args
)
```

VB

```
Function ReadAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    type As Type,
    args As Integer()
) As Object
```

Parameters

indexGroup	Type: System.UInt32 Index group of the ADS variable.
indexOffset	Type: System.UInt32 Index offset of the ADS variable.
type	Type: System.Type Type of the object to be read.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Object
The object the read data is written to.

Remarks

If the Type of the object to be read is a string type, the first element of the parameter args specifies the number of characters of the string. If the Type of the object to be read is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Reference

IAdsAnyAccess Interface [▶ 458]

ReadAny Overload [▶ 460]

TwinCAT.Ads Namespace [▶ 108]

5.2.38.1.2 IAdsAnyAccess.ReadAnyString Method

Overload List

	Name	Description
	ReadAnyString(Int32, Int32, Encoding) [▶ 464]	Reads the string.
	ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 465]	Reads the string

Reference

IAdsAnyAccess Interface [▶ 458]

TwinCAT.Ads Namespace [▶ 108]

IAdsAnyAccess.ReadAnyString Method (Int32, Int32, Encoding)

Reads the string.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string ReadAnyString(
    int variableHandle,
    int len,
    Encoding encoding
)
```

VB

```
Function ReadAnyString (
    variableHandle As Integer,
    len As Integer,
    encoding As Encoding
) As String
```

Parameters

variableHandle	Type: System.Int32 The variable handle.
len	Type: System.Int32 The length.
encoding	Type: System.Text.Encoding The encoding.

Return Value

Type: String
System.String.

Reference

IAdsAnyAccess Interface [▶ 458]

ReadAnyString Overload [▶ 464]

TwinCAT.Ads Namespace [▶ 108]

IAdsAnyAccess.ReadAnyString Method (UInt32, UInt32, Int32, Encoding)

Reads the string

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string ReadAnyString(
    uint indexGroup,
    uint indexOffset,
    int len,
    Encoding encoding
)
```

VB

```
Function ReadAnyString (
    indexGroup As UInteger,
    indexOffset As UInteger,
    len As Integer,
    encoding As Encoding
) As String
```

Parameters

indexGroup	Type: System.UInt32 The index group.
indexOffset	Type: System.UInt32 The index offset.

len	Type: System.Int32 The length.
encoding	Type: System.Text.Encoding The encoding.

Return Value

Type: String
System.String.

Remarks

ASCII Encoding expected

Reference

IAdsAnyAccess Interface [▶ 458]

ReadAnyString Overload [▶ 464]

TwinCAT.Ads Namespace [▶ 108]

5.2.38.1.3 IAdsAnyAccess.WriteAny Method

Overload List

	Name	Description
	WriteAny(Int32, Object) [▶ 466]	Writes an object synchronously to an ADS device.
	WriteAny(Int32, Object, .Int32.) [▶ 467]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.
	WriteAny(UInt32, UInt32, Object) [▶ 468]	Writes an object synchronously to an ADS device.
	WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 468]	Writes an object synchronously to an ADS device.

Reference

IAdsAnyAccess Interface [▶ 458]

TwinCAT.Ads Namespace [▶ 108]

IAdsAnyAccess.WriteAny Method (Int32, Object)

Writes an object synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void WriteAny(
    int variableHandle,
    Object value
)
```

VB

```
Sub WriteAny (
    variableHandle As Integer,
    value As Object
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable.
value	Type: System.Object Object to write to the ADS device.

Reference

IAdsAnyAccess Interface [▶ 458]

WriteAny Overload [▶ 466]

TwinCAT.Ads Namespace [▶ 108]

IAdsAnyAccess.WriteAny Method (Int32, Object, .Int32.)

Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void WriteAny(
    int variableHandle,
    Object value,
    int[] args
)
```

VB

```
Sub WriteAny (
    variableHandle As Integer,
    value As Object,
    args As Integer()
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable.
value	Type: System.Object Object to write to the ADS device.

args	Type: .System.Int32. Additional arguments.

Reference

IAdsAnyAccess Interface [▶ 458]

WriteAny Overload [▶ 466]

TwinCAT.Ads Namespace [▶ 108]

IAdsAnyAccess.WriteAny Method (UInt32, UInt32, Object)

Writes an object synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void WriteAny(
    uint indexGroup,
    uint indexOffset,
    Object value
)
```

VB

```
Sub WriteAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    value As Object
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
value	Type: System.Object Object to write to the ADS device.

Reference

IAdsAnyAccess Interface [▶ 458]

WriteAny Overload [▶ 466]

TwinCAT.Ads Namespace [▶ 108]

IAdsAnyAccess.WriteAny Method (UInt32, UInt32, Object, .Int32.)

Writes an object synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void WriteAny(
    uint indexGroup,
    uint indexOffset,
    Object value,
    int[] args
)
```

VB

```
Sub WriteAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    value As Object,
    args As Integer()
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
value	Type: System.Object Object to write to the ADS device.
args	Type: .System.Int32. Additional arguments.

Remarks

If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.

Reference

IAdsAnyAccess Interface [▶ 458]

WriteAny Overload [▶ 466]

TwinCAT.Ads Namespace [▶ 108]

5.2.39 IAdsConnection Interface

ADS Connection interface

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IAdsConnection : IConnection,
    IConnectionStateProvider, IAdsNotifications, IAdsAnyAccess, IAdsHandleAccess, ITcAdsRpcInvoke
```

VB

```
Public Interface IAdsConnection
    Inherits IConnection, IConnectionStateProvider, IAdsNotifications, IAdsAnyAccess,
    IAdsHandleAccess, ITcAdsRpcInvoke
```

The IAdsConnection type exposes the following members.

Properties

	Name	Description
	Address [▶ 477]	Gets the AmsAddress [▶ 410] of the ADS server.
	ClientAddress [▶ 477]	Get the AmsAddress [▶ 410] of the ADS client.
	ConnectionState [▶ 53]	Gets the current Connection state of the IConnectionStateProvider [▶ 52] (Inherited from IConnectionStateProvider [▶ 52].)
	Id [▶ 48]	Gets the Connection Identifier . (Inherited from IConnection [▶ 46].)
	IsConnected [▶ 48]	Gets a value indicating whether the local ADS port was opened successfully. It does not indicate if the target port is available. Use the method ReadState to determine if the target port is available. (Inherited from IConnection [▶ 46].)
	IsLocal [▶ 477]	Gets a value indicating whether the ADS client is connected to a ADS Server on the local computer.
	Session [▶ 49]	Gets the session that initiated this IConnection [▶ 46] (Inherited from IConnection [▶ 46].)
	Timeout [▶ 49]	Gets the timeout (in milliseconds) (Inherited from IConnection [▶ 46].)

Methods

	Name	Description
	AddDeviceNotification(String, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 500]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 501]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 502]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32,	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)

	Name	Description
	Int32, AdsTransMode, Int32, Int32, Object) [▶ 504]	
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type) [▶ 506]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 507]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type) [▶ 508]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 509]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported. (Inherited from IAdsNotifications [▶ 496].)
	Close [▶ 50]	Closes this IConnection [▶ 46] (Inherited from IConnection [▶ 46].)
	Connect [▶ 50]	(Re)Connects the IConnection [▶ 46] when disconnected. (Inherited from IConnection [▶ 46].)
	CreateVariableHandle [▶ 486]	Generates a unique handle for an ADS variable. (Inherited from IAdsHandleAccess [▶ 485].)
	DeleteDeviceNotification [▶ 510]	Deletes an existing notification. (Inherited from IAdsNotifications [▶ 496].)
	DeleteVariableHandle [▶ 487]	Releases the handle of a ADS variable again. (Inherited from IAdsHandleAccess [▶ 485].)
	Disconnect [▶ 51]	Disconnects this IConnection [▶ 46]. (Inherited from IConnection [▶ 46].)
	InvokeRpcMethod(String, Int32, .Object.) [▶ 537]	Invokes the specified RPC Method (Inherited from ITcAdsRpcInvoke [▶ 534].)
	InvokeRpcMethod(String, String, .Object.) [▶ 538]	Invokes the specified RPC Method (Inherited from ITcAdsRpcInvoke [▶ 534].)

	Name	Description
	InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 539]	Invokes the specified RPC Method (Inherited from ITcAdsRpcInvoke [▶ 534].)
	InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 540]	Invokes the specified RPC Method. (Inherited from ITcAdsRpcInvoke [▶ 534].)
	Read(Int32, AdsStream) [▶ 488]	Reads data synchronously from an ADS device and writes it to the given stream. (Inherited from IAdsHandleAccess [▶ 485].)
	Read(Int32, AdsStream, Int32, Int32) [▶ 489]	Reads data synchronously from an ADS device and writes it to the given stream. (Inherited from IAdsHandleAccess [▶ 485].)
	ReadAny(Int32, Type) [▶ 461]	Reads data synchronously from an ADS device and writes it to an object. (Inherited from IAdsAnyAccess [▶ 458].)
	ReadAny(Int32, Type, .Int32.) [▶ 461]	Reads data synchronously from an ADS device and writes it to an object. (Inherited from IAdsAnyAccess [▶ 458].)
	ReadAny(UInt32, UInt32, Type) [▶ 462]	Reads data synchronously from an ADS device and writes it to an object. (Inherited from IAdsAnyAccess [▶ 458].)
	ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 463]	Reads data synchronously from an ADS device and writes it to an object. (Inherited from IAdsAnyAccess [▶ 458].)
	ReadAnyString(Int32, Int32, Encoding) [▶ 464]	Reads the string. (Inherited from IAdsAnyAccess [▶ 458].)
	ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 465]	Reads the string (Inherited from IAdsAnyAccess [▶ 458].)
	ReadDeviceInfo [▶ 483]	Reads the identification and version number of an ADS server.
	ReadWrite [▶ 490]	Writes data synchronously to an ADS device and then Reads data from this device. (Inherited from IAdsHandleAccess [▶ 485].)
	TryAddDeviceNotification [▶ 511]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	TryAddDeviceNotificationEx [▶ 512]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	TryDeleteDeviceNotification [▶ 513]	Deletes an existing notification. (Inherited from IAdsNotifications [▶ 496].)
	TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 541]	Invokes the specified RPC Method (Inherited from ITcAdsRpcInvoke [▶ 534].)

	Name	Description
	TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 542]	Invokes the specified RPC Method (Inherited from ITcAdsRpcInvoke [▶ 534].)
	TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 543]	Invokes the specified RPC Method (Inherited from ITcAdsRpcInvoke [▶ 534].)
	TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 544]	Invokes the specified RPC Method. (Inherited from ITcAdsRpcInvoke [▶ 534].)
	TryRead [▶ 491]	Reads data synchronously from an ADS device and writes it to the given stream. (Inherited from IAdsHandleAccess [▶ 485].)
	TryReadWrite [▶ 492]	Writes data synchronously to an ADS device and then Reads data from this device. (Inherited from IAdsHandleAccess [▶ 485].)
	TryWrite [▶ 493]	Writes data synchronously to an ADS device. (Inherited from IAdsHandleAccess [▶ 485].)
	Write(Int32, AdsStream) [▶ 494]	Writes data synchronously to an ADS device. (Inherited from IAdsHandleAccess [▶ 485].)
	Write(Int32, AdsStream, Int32, Int32) [▶ 495]	Writes data synchronously to an ADS device. (Inherited from IAdsHandleAccess [▶ 485].)
	WriteAny(Int32, Object) [▶ 466]	Writes an object synchronously to an ADS device. (Inherited from IAdsAnyAccess [▶ 458].)
	WriteAny(Int32, Object, .Int32.) [▶ 467]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string. (Inherited from IAdsAnyAccess [▶ 458].)
	WriteAny(UInt32, UInt32, Object) [▶ 468]	Writes an object synchronously to an ADS device. (Inherited from IAdsAnyAccess [▶ 458].)
	WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 468]	Writes an object synchronously to an ADS device. (Inherited from IAdsAnyAccess [▶ 458].)

Events

	Name	Description
	AdsNotification [▶ 514]	Occurs when the ADS device sends a notification to the client. (Inherited from IAdsNotifications [▶ 496].)
	AdsNotificationError [▶ 515]	Occurs when a exception has occurred during notification management. (Inherited from IAdsNotifications [▶ 496].)
	AdsNotificationEx [▶ 516]	Occurs when the ADS devices sends a notification to the client. (Inherited from IAdsNotifications [▶ 496].)

	Name	Description
	ConnectionStateChanged [▶ 54]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed. (Inherited from IConnectionStateProvider [▶ 52].)

Extension Methods

	Name	Description
	PollAdsState(IObservable.Unit.) [▶ 825]	Overloaded. Gets an observable sequence of AdsState [▶ 365]s via Polling. (Defined by AdsClientExtensions [▶ 821].)
	PollAdsState(TimeSpan) [▶ 826]	Overloaded. Gets an observable sequence of AdsState [▶ 365]s via Polling. (Defined by AdsClientExtensions [▶ 821].)
	PollValues(String, Type, IObservable.Unit.) [▶ 849]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, TimeSpan) [▶ 850]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., TimeSpan) [▶ 853]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, IObservable.Unit., Func.Exception, Object.) [▶ 854]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, TimeSpan, Func.Exception, Object.) [▶ 855]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., IObservable.Unit., Func.Exception, Object.) [▶ 856]	Overloaded. Polls the symbol values on time points where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., TimeSpan, Func.Exception, Object.) [▶ 858]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, IObservable.Unit.) [▶ 842]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, TimeSpan) [▶ 843]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)

	Name	Description
	PollValues.T.(String, IObservable.Unit., Func.Exception, T.) [▶ 847]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, TimeSpan, Func.Exception, T.) [▶ 848]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T. (String, .Int32., IObservable.Unit.) [▶ 844]	Overloaded. Polls the symbol values on time points where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T. (String, .Int32., TimeSpan) [▶ 845]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T. (String, .Int32., IObservable.Unit., Func.Exception, T.) [▶ 851]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T. (String, .Int32., TimeSpan, Func.Exception, T.) [▶ 852]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	WhenNotification(ISymbol) [▶ 829]	Overloaded. Gets an observable sequence of Notification [▶ 862]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbolCollection) [▶ 830]	Overloaded. Gets an observable sequence of Notification [▶ 862] objects. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbol, NotificationSettings) [▶ 832]	Overloaded. Gets an observable sequence of Notification [▶ 862]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbolCollection, NotificationSettings) [▶ 833]	Overloaded. Gets an observable sequence of Notification [▶ 862] objects. (Defined by AdsClientExtensions [▶ 821].)
	WhenValueChanged [▶ 886]	Observable sequence of Values driven by ADS Notifications on the specified symbol. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues.T. (String, IObservable.T.) [▶ 859]	Overloaded. Writes the sequence of values to the symbol specified by the instance path. (Defined by AnyTypeExtensions [▶ 835].)

	Name	Description
	WriteValues.T. (String, IObservable.T., Action.Exception.) [▶ 860]	Overloaded. Writes the sequence of values to the symbol specified by the instance path. (Defined by AnyTypeExtensions [▶ 835].)

Remarks

The ADS connection interface represents the the logical point-to-point exception between Client and Server within an ADS Session.

Reference

TwinCAT.Ads Namespace [▶ 108]

TwinCAT.IConnection [▶ 46]

TwinCAT.Ads.IAdsAnyAccess [▶ 458]

TwinCAT.Ads.IAdsHandleAccess [▶ 485]

5.2.39.1 IAdsConnection Properties

The IAdsConnection [▶ 469] type exposes the following members.

Properties

	Name	Description
	Address [▶ 477]	Gets the AmsAddress [▶ 410] of the ADS server.
	ClientAddress [▶ 477]	Get the AmsAddress [▶ 410] of the ADS client.
	ConnectionState [▶ 53]	Gets the current Connection state of the IConnectionStateProvider [▶ 52] (Inherited from IConnectionStateProvider [▶ 52].)
	Id [▶ 48]	Gets the Connection Identifier . (Inherited from IConnection [▶ 46].)
	IsConnected [▶ 48]	Gets a value indicating whether the local ADS port was opened successfully. It does not indicate if the target port is available. Use the method ReadState to determine if the target port is available. (Inherited from IConnection [▶ 46].)
	IsLocal [▶ 477]	Gets a value indicating whether the ADS client is connected to a ADS Server on the local computer.
	Session [▶ 49]	Gets the session that initiated this IConnection [▶ 46] (Inherited from IConnection [▶ 46].)
	Timeout [▶ 49]	Gets the timeout (in milliseconds) (Inherited from IConnection [▶ 46].)

Reference

IAdsConnection Interface [▶ 469]

TwinCAT.Ads Namespace [▶ 108]

5.2.39.1.1 IAdsConnection.Address Property

Gets the AmsAddress [▶ 410] of the ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AmsAddress Address { get; }
```

VB

```
ReadOnly Property Address As AmsAddress
    Get
```

Property Value

Type: AmsAddress [▶ 410]

Reference

IAdsConnection Interface [▶ 469]

TwinCAT.Ads Namespace [▶ 108]

5.2.39.1.2 IAdsConnection.ClientAddress Property

Get the AmsAddress [▶ 410] of the ADS client.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AmsAddress ClientAddress { get; }
```

VB

```
ReadOnly Property ClientAddress As AmsAddress
    Get
```

Property Value

Type: AmsAddress [▶ 410]

Reference

IAdsConnection Interface [▶ 469]

TwinCAT.Ads Namespace [▶ 108]

5.2.39.1.3 IAdsConnection.IsLocal Property

Gets a value indicating whether the ADS client is connected to a ADS Server on the local computer.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsLocal { get; }
```

VB

```
ReadOnly Property IsLocal As Boolean
    Get
```

Property Value

Type: Boolean

Reference

IAdsConnection Interface [▶ 469]

TwinCAT.Ads Namespace [▶ 108]

5.2.39.2 IAdsConnection Methods

The IAdsConnection [▶ 469] type exposes the following members.

Methods

	Name	Description
	AddDeviceNotification(String, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 500]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 501]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 502]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32,	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)

	Name	Description
	Int32, AdsTransMode, Int32, Int32, Object) [▶ 504]	
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type) [▶ 506]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 507]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type) [▶ 508]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 509]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported. (Inherited from IAdsNotifications [▶ 496].)
	Close [▶ 50]	Closes this IConnection [▶ 46] (Inherited from IConnection [▶ 46].)
	Connect [▶ 50]	(Re)Connects the IConnection [▶ 46] when disconnected. (Inherited from IConnection [▶ 46].)
	CreateVariableHandle [▶ 486]	Generates a unique handle for an ADS variable. (Inherited from IAdsHandleAccess [▶ 485].)
	DeleteDeviceNotification [▶ 510]	Deletes an existing notification. (Inherited from IAdsNotifications [▶ 496].)
	DeleteVariableHandle [▶ 487]	Releases the handle of a ADS variable again. (Inherited from IAdsHandleAccess [▶ 485].)
	Disconnect [▶ 51]	Disconnects this IConnection [▶ 46]. (Inherited from IConnection [▶ 46].)
	InvokeRpcMethod(String, Int32, .Object.) [▶ 537]	Invokes the specified RPC Method (Inherited from ITcAdsRpcInvoke [▶ 534].)
	InvokeRpcMethod(String, String, .Object.) [▶ 538]	Invokes the specified RPC Method (Inherited from ITcAdsRpcInvoke [▶ 534].)

	Name	Description
	InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 539]	Invokes the specified RPC Method (Inherited from ITcAdsRpcInvoke [▶ 534].)
	InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 540]	Invokes the specified RPC Method. (Inherited from ITcAdsRpcInvoke [▶ 534].)
	Read(Int32, AdsStream) [▶ 488]	Reads data synchronously from an ADS device and writes it to the given stream. (Inherited from IAdsHandleAccess [▶ 485].)
	Read(Int32, AdsStream, Int32, Int32) [▶ 489]	Reads data synchronously from an ADS device and writes it to the given stream. (Inherited from IAdsHandleAccess [▶ 485].)
	ReadAny(Int32, Type) [▶ 461]	Reads data synchronously from an ADS device and writes it to an object. (Inherited from IAdsAnyAccess [▶ 458].)
	ReadAny(Int32, Type, .Int32.) [▶ 461]	Reads data synchronously from an ADS device and writes it to an object. (Inherited from IAdsAnyAccess [▶ 458].)
	ReadAny(UInt32, UInt32, Type) [▶ 462]	Reads data synchronously from an ADS device and writes it to an object. (Inherited from IAdsAnyAccess [▶ 458].)
	ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 463]	Reads data synchronously from an ADS device and writes it to an object. (Inherited from IAdsAnyAccess [▶ 458].)
	ReadAnyString(Int32, Int32, Encoding) [▶ 464]	Reads the string. (Inherited from IAdsAnyAccess [▶ 458].)
	ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 465]	Reads the string (Inherited from IAdsAnyAccess [▶ 458].)
	ReadDeviceInfo [▶ 483]	Reads the identification and version number of an ADS server.
	ReadWrite [▶ 490]	Writes data synchronously to an ADS device and then Reads data from this device. (Inherited from IAdsHandleAccess [▶ 485].)
	TryAddDeviceNotification [▶ 511]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	TryAddDeviceNotificationEx [▶ 512]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. (Inherited from IAdsNotifications [▶ 496].)
	TryDeleteDeviceNotification [▶ 513]	Deletes an existing notification. (Inherited from IAdsNotifications [▶ 496].)
	TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 541]	Invokes the specified RPC Method (Inherited from ITcAdsRpcInvoke [▶ 534].)

	Name	Description
	TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 542]	Invokes the specified RPC Method (Inherited from ITcAdsRpcInvoke [▶ 534].)
	TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 543]	Invokes the specified RPC Method (Inherited from ITcAdsRpcInvoke [▶ 534].)
	TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 544]	Invokes the specified RPC Method. (Inherited from ITcAdsRpcInvoke [▶ 534].)
	TryRead [▶ 491]	Reads data synchronously from an ADS device and writes it to the given stream. (Inherited from IAdsHandleAccess [▶ 485].)
	TryReadWrite [▶ 492]	Writes data synchronously to an ADS device and then Reads data from this device. (Inherited from IAdsHandleAccess [▶ 485].)
	TryWrite [▶ 493]	Writes data synchronously to an ADS device. (Inherited from IAdsHandleAccess [▶ 485].)
	Write(Int32, AdsStream) [▶ 494]	Writes data synchronously to an ADS device. (Inherited from IAdsHandleAccess [▶ 485].)
	Write(Int32, AdsStream, Int32, Int32) [▶ 495]	Writes data synchronously to an ADS device. (Inherited from IAdsHandleAccess [▶ 485].)
	WriteAny(Int32, Object) [▶ 466]	Writes an object synchronously to an ADS device. (Inherited from IAdsAnyAccess [▶ 458].)
	WriteAny(Int32, Object, .Int32.) [▶ 467]	Writes an object synchronously to an ADS device. If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string. (Inherited from IAdsAnyAccess [▶ 458].)
	WriteAny(UInt32, UInt32, Object) [▶ 468]	Writes an object synchronously to an ADS device. (Inherited from IAdsAnyAccess [▶ 458].)
	WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 468]	Writes an object synchronously to an ADS device. (Inherited from IAdsAnyAccess [▶ 458].)

Extension Methods

	Name	Description
	PollAdsState(IObservable.Unit.) [▶ 825]	Overloaded. Gets an observable sequence of AdsState [▶ 365]s via Polling. (Defined by AdsClientExtensions [▶ 821].)
	PollAdsState(TimeSpan) [▶ 826]	Overloaded. Gets an observable sequence of AdsState [▶ 365]s via Polling. (Defined by AdsClientExtensions [▶ 821].)

	Name	Description
	PollValues(String, Type, IObservable.Unit.) [▶ 849]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, TimeSpan) [▶ 850]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., TimeSpan) [▶ 853]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, IObservable.Unit., Func.Exception, Object.) [▶ 854]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, TimeSpan, Func.Exception, Object.) [▶ 855]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., IObservable.Unit., Func.Exception, Object.) [▶ 856]	Overloaded. Polls the symbol values on time points where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., TimeSpan, Func.Exception, Object.) [▶ 858]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, IObservable.Unit.) [▶ 842]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, TimeSpan) [▶ 843]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, IObservable.Unit., Func.Exception, T.) [▶ 847]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, TimeSpan, Func.Exception, T.) [▶ 848]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, .Int32., IObservable.Unit.) [▶ 844]	Overloaded. Polls the symbol values on time points where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, .Int32., TimeSpan) [▶ 845]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)

	Name	Description
	PollValues.T. (String, .Int32., IObservable.Unit., Func.Exception, T.) [▶ 851]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T. (String, .Int32., TimeSpan, Func.Exception, T.) [▶ 852]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	WhenNotification(ISymbol) [▶ 829]	Overloaded. Gets an observable sequence of Notification [▶ 862]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbolCollection) [▶ 830]	Overloaded. Gets an observable sequence of Notification [▶ 862] objects. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbol, NotificationSettings) [▶ 832]	Overloaded. Gets an observable sequence of Notification [▶ 862]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbolCollection, NotificationSettings) [▶ 833]	Overloaded. Gets an observable sequence of Notification [▶ 862] objects. (Defined by AdsClientExtensions [▶ 821].)
	WhenValueChanged [▶ 886]	Observable sequence of Values driven by ADS Notifications on the specified symbol. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues.T. (String, IObservable.T.) [▶ 859]	Overloaded. Writes the sequence of values to the symbol specified by the instance path. (Defined by AnyTypeExtensions [▶ 835].)
	WriteValues.T. (String, IObservable.T., Action.Exception.) [▶ 860]	Overloaded. Writes the sequence of values to the symbol specified by the instance path. (Defined by AnyTypeExtensions [▶ 835].)

Reference

IAdsConnection Interface [▶ 469]

TwinCAT.Ads Namespace [▶ 108]

5.2.39.2.1 IAdsConnection.ReadDeviceInfo Method

Reads the identification and version number of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
DeviceInfo ReadDeviceInfo()
```

VB

```
Function ReadDeviceInfo As DeviceInfo
```

Return Value

Type: DeviceInfo [▶ 455]
DeviceInfo struct containing the name of the device and the version information.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsConnection Interface [▶ 469]

TwinCAT.Ads Namespace [▶ 108]

5.2.39.3 IAdsConnection Events

The IAdsConnection [▶ 469] type exposes the following members.

Events

	Name	Description
	AdsNotification [▶ 514]	Occurs when the ADS device sends a notification to the client. (Inherited from IAdsNotifications [▶ 496].)
	AdsNotificationError [▶ 515]	Occurs when a exception has occurred during notification management. (Inherited from IAdsNotifications [▶ 496].)
	AdsNotificationEx [▶ 516]	Occurs when the ADS devices sends a notification to the client. (Inherited from IAdsNotifications [▶ 496].)
	ConnectionStateChanged [▶ 54]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed. (Inherited from IConnectionStateProvider [▶ 52].)

Reference

IAdsConnection Interface [▶ 469]

TwinCAT.Ads Namespace [▶ 108]

5.2.40 IAdsHandleAccess Interface

Interface for ads access via variable handle

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IAdsHandleAccess
```

VB

```
Public Interface IAdsHandleAccess
```

The IAdsHandleAccess type exposes the following members.

Methods

	Name	Description
	CreateVariableHandle [▶ 486]	Generates a unique handle for an ADS variable.
	DeleteVariableHandle [▶ 487]	Releases the handle of a ADS variable again.
	Read(Int32, AdsStream) [▶ 488]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(Int32, AdsStream, Int32, Int32) [▶ 489]	Reads data synchronously from an ADS device and writes it to the given stream.
	ReadWrite [▶ 490]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryRead [▶ 491]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryReadWrite [▶ 492]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryWrite [▶ 493]	Writes data synchronously to an ADS device.
	Write(Int32, AdsStream) [▶ 494]	Writes data synchronously to an ADS device.
	Write(Int32, AdsStream, Int32, Int32) [▶ 495]	Writes data synchronously to an ADS device.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.40.1 IAdsHandleAccess Methods

The IAdsHandleAccess [▶ 485] type exposes the following members.

Methods

	Name	Description
	CreateVariableHandle [▶ 486]	Generates a unique handle for an ADS variable.
	DeleteVariableHandle [▶ 487]	Releases the handle of a ADS variable again.
	Read(Int32, AdsStream) [▶ 488]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(Int32, AdsStream, Int32, Int32) [▶ 489]	Reads data synchronously from an ADS device and writes it to the given stream.
	ReadWrite [▶ 490]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryRead [▶ 491]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryReadWrite [▶ 492]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryWrite [▶ 493]	Writes data synchronously to an ADS device.
	Write(Int32, AdsStream) [▶ 494]	Writes data synchronously to an ADS device.
	Write(Int32, AdsStream, Int32, Int32) [▶ 495]	Writes data synchronously to an ADS device.

Reference

IAdsHandleAccess Interface [▶ 485]

TwinCAT.Ads Namespace [▶ 108]

5.2.40.1.1 IAdsHandleAccess.CreateVariableHandle Method

Generates a unique handle for an ADS variable.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int CreateVariableHandle(
    string variableName
)
```

VB

```
Function CreateVariableHandle (
    variableName As String
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable

Return Value

Type: Int32
The handle of the ADS Variable.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsHandleAccess Interface [▶ 485]

TwinCAT.Ads Namespace [▶ 108]

5.2.40.1.2 IAdsHandleAccess.DeleteVariableHandle Method

Releases the handle of a ADS variable again.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void DeleteVariableHandle(
    int variableHandle
)
```

VB

```
Sub DeleteVariableHandle (
    variableHandle As Integer
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsHandleAccess Interface [▶ 485]

TwinCAT.Ads Namespace [▶ 108]

5.2.40.1.3 IAdsHandleAccess.Read Method

Overload List

	Name	Description
	Read(Int32, AdsStream) [▶ 488]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(Int32, AdsStream, Int32, Int32) [▶ 489]	Reads data synchronously from an ADS device and writes it to the given stream.

Reference

IAdsHandleAccess Interface [▶ 485]

TwinCAT.Ads Namespace [▶ 108]

IAdsHandleAccess.Read Method (Int32, AdsStream)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Read(
    int variableHandle,
    AdsStream dataStream
)
```

VB

```
Function Read (
    variableHandle As Integer,
    dataStream As AdsStream
) As Integer
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.

Return Value

Type: Int32
Number of successfully returned data bytes.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsHandleAccess Interface [▶ 485]

Read Overload [▶ 488]

TwinCAT.Ads Namespace [▶ 108]

IAdsHandleAccess.Read Method (Int32, AdsStream, Int32, Int32)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Read(
    int variableHandle,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Function Read (
    variableHandle As Integer,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
) As Integer
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsHandleAccess Interface [▶ 485]

Read Overload [▶ 488]

TwinCAT.Ads Namespace [▶ 108]

5.2.40.1.4 IAdsHandleAccess.ReadWrite Method

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int ReadWrite(
    int variableHandle,
    AdsStream rdDataStream,
    int rdOffset,
    int rdLength,
    AdsStream wrDataStream,
    int wrOffset,
    int wrLength
)
```

VB

```
Function ReadWrite (
    variableHandle As Integer,
    rdDataStream As AdsStream,
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As AdsStream,
    wrOffset As Integer,
    wrLength As Integer
) As Integer
```

Parameters

variableHandle	Type: System.Int32 Variable handle.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsHandleAccess Interface [▶ 485]

TwinCAT.Ads Namespace [▶ 108]

5.2.40.1.5 IAdsHandleAccess.TryRead Method

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsErrorCode TryRead(
    int variableHandle,
    AdsStream dataStream,
    int offset,
    int length,
    out int readBytes
)
```

VB

```
Function TryRead (
    variableHandle As Integer,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
readBytes	Type: System.Int32. Number of successfully returned data bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode

Exceptions

Exception	Condition
ArgumentException	

Reference

IAdsHandleAccess Interface [▶ 485]

TwinCAT.Ads Namespace [▶ 108]

5.2.40.1.6 IAdsHandleAccess.TryReadWrite Method

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsErrorCode TryReadWrite(
    int variableHandle,
    AdsStream rdDataStream,
    int rdOffset,
    int rdLength,
    AdsStream wrDataStream,
    int wrOffset,
    int wrLength,
    out int readBytes
)
```

VB

```
Function TryReadWrite (
    variableHandle As Integer,
    rdDataStream As AdsStream,
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As AdsStream,
    wrOffset As Integer,
    wrLength As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

variableHandle	Type: System.Int32 Variable handle.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ArgumentException	

Reference

IAdsHandleAccess Interface [▶ 485]

TwinCAT.Ads Namespace [▶ 108]

5.2.40.1.7 IAdsHandleAccess.TryWrite Method

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsErrorCode TryWrite(
    int variableHandle,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Function TryWrite (
    variableHandle As Integer,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
) As AdsErrorCode
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ArgumentException	

Reference

IAdsHandleAccess Interface [▶ 485]

TwinCAT.Ads Namespace [▶ 108]

5.2.40.1.8 IAdsHandleAccess.Write Method

Overload List

	Name	Description
	Write(Int32, AdsStream) [▶ 494]	Writes data synchronously to an ADS device.
	Write(Int32, AdsStream, Int32, Int32) [▶ 495]	Writes data synchronously to an ADS device.

Reference

IAdsHandleAccess Interface [▶ 485]

TwinCAT.Ads Namespace [▶ 108]

IAdsHandleAccess.Write Method (Int32, AdsStream)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void Write(
    int variableHandle,
    AdsStream dataStream
)
```

VB

```
Sub Write (
    variableHandle As Integer,
    dataStream As AdsStream
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsHandleAccess Interface [▶ 485]

Write Overload [▶ 494]

TwinCAT.Ads Namespace [▶ 108]

IAdsHandleAccess.Write Method (Int32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void Write(
    int variableHandle,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Sub Write (
    variableHandle As Integer,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsHandleAccess Interface [▶ 485]

Write Overload [▶ 494]

TwinCAT.Ads Namespace [▶ 108]

5.2.41 IAdsNotifications Interface

Interface for Notification management.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IAdsNotifications
```

VB

```
Public Interface IAdsNotifications
```

The IAdsNotifications type exposes the following members.

Methods

	Name	Description
	AddDeviceNotification(String, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 500]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 501]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 502]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 504]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String,	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AdsTransMode, Int32, Int32, Object, Type) [▶ 506]	
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 507]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type) [▶ 508]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 509]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	DeleteDeviceNotification [▶ 510]	Deletes an existing notification.
	TryAddDeviceNotification [▶ 511]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	TryAddDeviceNotificationEx [▶ 512]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	TryDeleteDeviceNotification [▶ 513]	Deletes an existing notification.

Events

	Name	Description
	AdsNotification [▶ 514]	Occurs when the ADS device sends a notification to the client.
	AdsNotificationError [▶ 515]	Occurs when a exception has occurred during notification management.
	AdsNotificationEx [▶ 516]	Occurs when the ADS devices sends a notification to the client.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.41.1 IAdsNotifications Methods

The IAdsNotifications [▶ 496] type exposes the following members.

Methods

	Name	Description
	AddDeviceNotification(String, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 500]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 501]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 502]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 504]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type) [▶ 506]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 507]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32,	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AdsTransMode, Int32, Int32, Object, Type) [▶ 508]	
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 509]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	DeleteDeviceNotification [▶ 510]	Deletes an existing notification.
	TryAddDeviceNotification [▶ 511]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	TryAddDeviceNotificationEx [▶ 512]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	TryDeleteDeviceNotification [▶ 513]	Deletes an existing notification.

Reference

IAdsNotifications Interface [▶ 496]

TwinCAT.Ads Namespace [▶ 108]

5.2.41.1.1 IAdsNotifications.AddDeviceNotification Method

Overload List

	Name	Description
	AddDeviceNotification(String, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 500]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 501]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String, AdsStream, Int32, Int32,	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AdsTransMode, Int32, Int32, Object) [▶ 502]	
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 504]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Reference

IAdsNotifications Interface [▶ 496]

TwinCAT.Ads Namespace [▶ 108]

IAdsNotifications.AddDeviceNotification Method (String, AdsStream, AdsTransMode, Int32, Int32, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int AddDeviceNotification(
    string variableName,
    AdsStream dataStream,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData
)
```

VB

```
Function AddDeviceNotification (
    variableName As String,
    dataStream As AdsStream,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.

cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsNotifications Interface [▶ 496]

AddDeviceNotification Overload [▶ 499]

TwinCAT.Ads Namespace [▶ 108]

IAdsNotifications.AddDeviceNotification Method (UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int AddDeviceNotification(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData
)
```

VB

```
Function AddDeviceNotification (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Reference

IAdsNotifications Interface [▶ 496]

AddDeviceNotification Overload [▶ 499]

TwinCAT.Ads Namespace [▶ 108]

IAdsNotifications.AddDeviceNotification Method (String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int AddDeviceNotification(
    string variableName,
    AdsStream dataStream,
    int offset,
    int length,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData
)
```

VB

```
Function AddDeviceNotification (
    variableName As String,
    dataStream As AdsStream,
    offset As Integer,
```

```
    length As Integer,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Remarks

AdsTransMode [▶ 401]	Parameter semantic
CyclicInContext [▶ 401]	Value of parameter is interpreted as task context number ContextMask [▶ 555]
OnChangeInContext [▶ 401]	Value of parameter is interpreted as task context number ContextMask [▶ 555]

Reference

IAdsNotifications Interface [▶ 496]

AddDeviceNotification Overload [▶ 499]

TwinCAT.Ads Namespace [▶ 108]

IAdsNotifications.AddDeviceNotification Method (UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int AddDeviceNotification(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData
)
```

VB

```
Function AddDeviceNotification (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsNotifications Interface [▶ 496]

AddDeviceNotification Overload [▶ 499]

TwinCAT.Ads Namespace [▶ 108]

5.2.41.1.2 IAdsNotifications.AddDeviceNotificationEx Method

Overload List

	Name	Description
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type) [▶ 506]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 507]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type) [▶ 508]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 509]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Reference

IAdsNotifications Interface [▶ 496]

TwinCAT.Ads Namespace [▶ 108]

IAdsNotifications.AddDeviceNotificationEx Method (String, AdsTransMode, Int32, Int32, Object, Type)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int AddDeviceNotificationEx(
    string variableName,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData,
    Type type
)
```

VB

```
Function AddDeviceNotificationEx (
    variableName As String,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object,
    type As Type
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsNotifications Interface [▶ 496]

AddDeviceNotificationEx Overload [▶ 505]

TwinCAT.Ads Namespace [▶ 108]

IAdsNotifications.AddDeviceNotificationEx Method (String, AdsTransMode, Int32, Int32, Object, Type, .Int32.)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int AddDeviceNotificationEx(
    string variableName,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData,
    Type type,
    int[] args
)
```

VB

```
Function AddDeviceNotificationEx (
    variableName As String,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object,
    type As Type,
    args As Integer()
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsNotifications Interface [▶ 496]

AddDeviceNotificationEx Overload [▶ 505]

TwinCAT.Ads Namespace [▶ 108]

IAdsNotifications.AddDeviceNotificationEx Method (UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int AddDeviceNotificationEx(
    uint indexGroup,
    uint indexOffset,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData,
    Type type
)
```

VB

```
Function AddDeviceNotificationEx (
    indexGroup As UInteger,
    indexOffset As UInteger,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object,
    type As Type
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.

cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsNotifications Interface [▶ 496]

AddDeviceNotificationEx Overload [▶ 505]

TwinCAT.Ads Namespace [▶ 108]

IAdsNotifications.AddDeviceNotificationEx Method (UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int AddDeviceNotificationEx(
    uint indexGroup,
    uint indexOffset,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData,
    Type type,
    int[] args
)
```

VB

```
Function AddDeviceNotificationEx (
    indexGroup As UInteger,
    indexOffset As UInteger,
    transMode As AdsTransMode,
    cycleTime As Integer,
```

```
    maxDelay As Integer,
    userData As Object,
    type As Type,
    args As Integer()
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsNotifications Interface [▶ 496]

AddDeviceNotificationEx Overload [▶ 505]

TwinCAT.Ads Namespace [▶ 108]

5.2.41.1.3 IAdsNotifications.DeleteDeviceNotification Method

Deletes an existing notification.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void DeleteDeviceNotification(
    int notificationHandle
)
```

VB

```
Sub DeleteDeviceNotification (
    notificationHandle As Integer
)
```

Parameters

notificationHandle	Type: System.Int32 Handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsNotifications Interface [▶ 496]

TwinCAT.Ads Namespace [▶ 108]

5.2.41.1.4 IAdsNotifications.TryAddDeviceNotification Method

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsErrorCode TryAddDeviceNotification(
    string variableName,
    AdsStream dataStream,
    int offset,
    int length,
    NotificationSettings settings,
    Object userData,
    out uint handle
)
```

VB

```
Function TryAddDeviceNotification (
    variableName As String,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    settings As NotificationSettings,
    userData As Object,
    <OutAttribute> ByRef handle As UInteger
) As AdsErrorCode
```

Parameters

variableName	Type: System.String Name of the ADS variable.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
settings	Type: TwinCAT.Ads.NotificationSettings [▶ 573] The settings.
userData	Type: System.Object This object can be used to store user specific data.
handle	Type: System.UInt32. The handle.

Return Value

Type: AdsErrorCode [▶ 305]
The ADS ErrorCode.

Reference

IAdsNotifications Interface [▶ 496]

TwinCAT.Ads Namespace [▶ 108]

5.2.41.1.5 IAdsNotifications.TryAddDeviceNotificationEx Method

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsErrorCode TryAddDeviceNotificationEx(
    string variableName,
    NotificationSettings settings,
    Object userData,
    Type type,
    int[] args,
    out uint handle
)
```

VB

```
Function TryAddDeviceNotificationEx (
    variableName As String,
    settings As NotificationSettings,
    userData As Object,
    type As Type,
    args As Integer(),
    <OutAttribute> ByRef handle As UInteger
) As AdsErrorCode
```

Parameters

variableName	Type: System.String Name of the ADS variable.
settings	Type: TwinCAT.Ads.NotificationSettings [▶ 573] The settings.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.
args	Type: .System.Int32. Additional arguments.
handle	Type: System.UInt32. The handle.

Return Value

Type: AdsErrorCode [▶ 305]
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsNotifications Interface [▶ 496]

TwinCAT.Ads Namespace [▶ 108]

5.2.41.1.6 IAdsNotifications.TryDeleteDeviceNotification Method

Deletes an existing notification.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsErrorCode TryDeleteDeviceNotification(
    uint notificationHandle
)
```

VB

```
Function TryDeleteDeviceNotification (
    notificationHandle As UInteger
) As AdsErrorCode
```

Parameters

notificationHandle	Type: System.UInt32 Handle of the notification.

Return Value

Type: AdsErrorCode [▶ 305]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

IAdsNotifications Interface [▶ 496]

TwinCAT.Ads Namespace [▶ 108]

5.2.41.2 IAdsNotifications Events

The IAdsNotifications [▶ 496] type exposes the following members.

Events

	Name	Description
	AdsNotification [▶ 514]	Occurs when the ADS device sends a notification to the client.
	AdsNotificationError [▶ 515]	Occurs when a exception has occurred during notification management.
	AdsNotificationEx [▶ 516]	Occurs when the ADS devices sends a notification to the client.

Reference

IAdsNotifications Interface [▶ 496]

TwinCAT.Ads Namespace [▶ 108]

5.2.41.2.1 IAdsNotifications.AdsNotification Event

Occurs when the ADS device sends a notification to the client.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
event AdsNotificationEventHandler AdsNotification
```

VB

```
Event AdsNotification As AdsNotificationEventHandler
```

Value

Type: TwinCAT.Ads.AdsNotificationEventHandler [▶ 342]

Examples

The following sample shows how to register/unregister for AdsNotification

Receive AdsNotifications

```
AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event handler
    client.AdsNotification += Client_AdsNotification;

    // Connect to target
    client.Connect("1.2.3.4.5.6", 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a DINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotification("MAIN.nCounter", readStream, AdsTransMode.
OnChange, 200, 0, null);
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotification -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationEventArgs e)
{
    int offset = (int)e.DataStream.Position;
    int length = (int)e.DataStream.Length;

    e.DataStream.Position = offset;
    AdsBinaryReader reader = new AdsBinaryReader(e.DataStream);

    // Read the Unmarshalled data
    //byte[] data = reader.ReadBytes(length);

    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = reader.ReadUInt32();
}
```

Reference

IAdsNotifications Interface [▶ 496]

TwinCAT.Ads Namespace [▶ 108]

5.2.41.2.2 IAdsNotifications.AdsNotificationError Event

Occurs when a exception has occurred during notification management.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
event AdsNotificationErrorEventHandler AdsNotificationError
```

VB

```
Event AdsNotificationError As AdsNotificationErrorEventHandler
```

Value

Type: TwinCAT.Ads.AdsNotificationErrorEventHandler [▶ 335]

Reference

IAdsNotifications Interface [▶ 496]

TwinCAT.Ads Namespace [▶ 108]

5.2.41.2.3 IAdsNotifications.AdsNotificationEx Event

Occurs when the ADS devices sends a notification to the client.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
event AdsNotificationExEventHandler AdsNotificationEx
```

VB

```
Event AdsNotificationEx As AdsNotificationExEventHandler
```

Value

Type: TwinCAT.Ads.AdsNotificationExEventHandler [▶ 346]

Reference

IAdsNotifications Interface [▶ 496]

TwinCAT.Ads Namespace [▶ 108]

5.2.42 IAdsSession Interface

Interface IAdsSession

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IAdsSession : ISession,
    IConnectionStateProvider
```

VB

```
Public Interface IAdsSession
    Inherits ISession, IConnectionStateProvider
```

The IAdsSession type exposes the following members.

Properties

	Name	Description
	Address [▶ 518]	Gets the Address specifier of the Session / connection
	AddressSpecifier [▶ 61]	Gets the communication endpoint address string representation. (Inherited from ISession [▶ 60].)
	Connection [▶ 62]	Gets the Connection object. (Inherited from ISession [▶ 60].)
	ConnectionState [▶ 53]	Gets the current Connection state of the IConnectionStateProvider [▶ 52] (Inherited from IConnectionStateProvider [▶ 52].)
	EstablishedAt [▶ 62]	Gets the UTC time when the session was established. (Inherited from ISession [▶ 60].)
	Id [▶ 63]	Gets the Session Id (Inherited from ISession [▶ 60].)
	IsConnected [▶ 63]	Gets a value indicating whether the session is connected. (Inherited from ISession [▶ 60].)
	NetId [▶ 519]	Gets the NetId of the Session
	Owner [▶ 519]	Gets the Session owner.
	Port [▶ 520]	Gets the Ams Port of the Session

Methods

	Name	Description
	Close [▶ 64]	Closes this ISession [▶ 60] (Inherited from ISession [▶ 60].)
	Connect [▶ 65]	Connects the session and returns the established IConnection [▶ 46] object. (Inherited from ISession [▶ 60].)
	Disconnect [▶ 65]	Disconnects the ISession [▶ 60] (Inherited from ISession [▶ 60].)

Events

	Name	Description
	ConnectionStateChanged [▶ 54]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed. (Inherited from IConnectionStateProvider [▶ 52].)

Reference

TwinCAT.Ads Namespace [▶ 108]

TwinCAT.ISession [▶ 60]

5.2.42.1 IAdsSession Properties

The IAdsSession [▶ 516] type exposes the following members.

Properties

	Name	Description
	Address [▶ 518]	Gets the Address specifier of the Session / connection
	AddressSpecifier [▶ 61]	Gets the communication endpoint address string representation. (Inherited from ISession [▶ 60].)
	Connection [▶ 62]	Gets the Connection object. (Inherited from ISession [▶ 60].)
	ConnectionState [▶ 53]	Gets the current Connection state of the IConnectionStateProvider [▶ 52] (Inherited from IConnectionStateProvider [▶ 52].)
	EstablishedAt [▶ 62]	Gets the UTC time when the session was established. (Inherited from ISession [▶ 60].)
	Id [▶ 63]	Gets the Session Id (Inherited from ISession [▶ 60].)
	IsConnected [▶ 63]	Gets a value indicating whether the session is connected. (Inherited from ISession [▶ 60].)
	NetId [▶ 519]	Gets the NetId of the Session
	Owner [▶ 519]	Gets the Session owner.
	Port [▶ 520]	Gets the Ams Port of the Session

Reference

IAdsSession Interface [▶ 516]

TwinCAT.Ads Namespace [▶ 108]

5.2.42.1.1 IAdsSession.Address Property

Gets the Address specifier of the Session / connection

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AmsAddress Address { get; }
```

VB

```
ReadOnly Property Address As AmsAddress
    Get
```

Property Value

Type: AmsAddress [▶ 410]
The address.

Reference

IAdsSession Interface [▶ 516]

TwinCAT.Ads Namespace [▶ 108]

5.2.42.1.2 IAdsSession.NetId Property

Gets the NetId of the Session

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AmsNetId NetId { get; }
```

VB

```
ReadOnly Property NetId As AmsNetId
    Get
```

Property Value

Type: AmsNetId [▶ 428]
The net identifier.

Reference

IAdsSession Interface [▶ 516]

TwinCAT.Ads Namespace [▶ 108]

5.2.42.1.3 IAdsSession.Owner Property

Gets the Session owner.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object Owner { get; }
```

VB

```
ReadOnly Property Owner As Object
    Get
```

Property Value

Type: Object
The owner or NULL

Reference

IAdsSession Interface [▶ 516]

TwinCAT.Ads Namespace [▶ 108]

5.2.42.1.4 IAdsSession.Port Property

Gets the Ams Port of the Session

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Port { get; }
```

VB

```
ReadOnly Property Port As Integer
    Get
```

Property Value

Type: Int32
The port.

Reference

IAdsSession Interface [▶ 516]

TwinCAT.Ads Namespace [▶ 108]

5.2.42.2 IAdsSession Methods

The IAdsSession [▶ 516] type exposes the following members.

Methods

	Name	Description
	Close [▶ 64]	Closes this ISession [▶ 60] (Inherited from ISession [▶ 60].)
	Connect [▶ 65]	Connects the session and returns the established IConnection [▶ 46] object. (Inherited from ISession [▶ 60].)
	Disconnect [▶ 65]	Disconnects the ISession [▶ 60] (Inherited from ISession [▶ 60].)

Reference

IAdsSession Interface [▶ 516]

TwinCAT.Ads Namespace [▶ 108]

5.2.42.3 IAdsSession Events

The IAdsSession [▶ 516] type exposes the following members.

Events

	Name	Description
	ConnectionStateChanged [▶ 54]	Occurs when connection status of the IConnectionStateProvider [▶ 52] has been changed. (Inherited from IConnectionStateProvider [▶ 52].)

	Name	Description

Reference

IAdsSession Interface [▶ 516]

TwinCAT.Ads Namespace [▶ 108]

5.2.43 IAdsSessionSettings Interface

Interface for ADS Session Settings

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IAdsSessionSettings : ISessionSettings
```

VB

```
Public Interface IAdsSessionSettings
    Inherits ISessionSettings
```

The IAdsSessionSettings type exposes the following members.

Properties

	Name	Description
	ResurrectionTime [▶ 522]	Gets or sets the resurrection time.
	SymbolLoader [▶ 522]	Gets or sets the symbol loader settings
	Timeout [▶ 523]	Gets the ADS timeout in milliseconds.

Reference

TwinCAT.Ads Namespace [▶ 108]

TwinCAT.ISessionSettings [▶ 66]

5.2.43.1 IAdsSessionSettings Properties

The IAdsSessionSettings [▶ 521] type exposes the following members.

Properties

	Name	Description
	ResurrectionTime [▶ 522]	Gets or sets the resurrection time.
	SymbolLoader [▶ 522]	Gets or sets the symbol loader settings

	Name	Description
	Timeout [▶ 523]	Gets the ADS timeout in milliseconds.

Reference

IAdsSessionSettings Interface [▶ 521]

TwinCAT.Ads Namespace [▶ 108]

5.2.43.1.1 IAdsSessionSettings.ResurrectionTime Property

Gets or sets the resurrection time.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
TimeSpan ResurrectionTime { get; set; }
```

VB

```
Property ResurrectionTime As TimeSpan
    Get
    Set
```

Property Value

Type: TimeSpan
The resurrection time.

Reference

IAdsSessionSettings Interface [▶ 521]

TwinCAT.Ads Namespace [▶ 108]

5.2.43.1.2 IAdsSessionSettings.SymbolLoader Property

Gets or sets the symbol loader settings

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
SymbolLoaderSettings SymbolLoader { get; set; }
```

VB

```
Property SymbolLoader As SymbolLoaderSettings
    Get
    Set
```

Property Value

Type: SymbolLoaderSettings [▶ 615]
The symbol loader.

Reference

IAdsSessionSettings Interface [▶ 521]

TwinCAT.Ads Namespace [▶ 108]

5.2.43.1.3 IAdsSessionSettings.Timeout Property

Gets the ADS timeout in milliseconds.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Timeout { get; }
```

VB

```
ReadOnly Property Timeout As Integer
    Get
```

Property Value

Type: Int32
The timeout.

Reference

IAdsSessionSettings Interface [▶ 521]

TwinCAT.Ads Namespace [▶ 108]

5.2.44 IFailFastHandler Interface

Interface for a fast failing (Circuit breaker) ads handler

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IFailFastHandler
```

VB

```
Public Interface IFailFastHandler
```

Remarks

If a target is not available it will throw Timeout exceptions after a Default time of 5 seconds. To prevent hanging applications and bring more robustness into the communication (less consumption of ADS Mailbox memory), a second try to call the target should fail fast - not waiting for the Timeout. Only after a dedicated reconnection timeout timespan, real communication should be retried.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.45 ITcAdsDataType Interface

Interface ITcAdsDataType

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ITcAdsDataType : IDataType,
    IBitSize
```

VB

```
Public Interface ITcAdsDataType
    Inherits IDataType, IBitSize
```

The ITcAdsDataType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 527]	Gets the Base Type of the Type (if enum, alias, array)
	BaseTypeName [▶ 527]	Gets the name of the base type (if enum, alias, array)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	DataTypeId [▶ 528]	Gets the data type identifier.
	Dimensions [▶ 528]	Gets the dimensions of an array type
	EnumValues [▶ 529]	Enumeration specification (if enum)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)

	Name	Description
	HasArrayInfo [▶ 529]	Gets a value indicating whether this instance has dimension information.
	HasEnumInfo [▶ 530]	Gets a value indicating whether this instance has enum information.
	HasRpcMethods [▶ 530]	Gets a value indicating whether this instance has RPC methods (Struct types only)
	HasSubItemInfo [▶ 531]	Gets a value indicating whether this instance has sub items.
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsJaggedArray [▶ 531]	Gets a value indicating whether this dataType is a jagged array.
	IsOversamplingArray [▶ 532]	Gets a value indicating whether this instance is an oversampling array.
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	IsSubItem [▶ 532]	Determines whether this object is an SubItem (DataType Member, true) or a native DataType (false)
	ManagedType [▶ 533]	Managed Type of the (simple,primitive) type.
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	RpcMethods [▶ 533]	Gets the RPC method descriptions
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubItems [▶ 534]	Get the SubElements of this ITcAdsDataType.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1 ITcAdsDataType Properties

The ITcAdsDataType [▶ 524] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 527]	Gets the Base Type of the Type (if enum, alias, array)
	BaseTypeName [▶ 527]	Gets the name of the base type (if enum, alias, array)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	DataTypeId [▶ 528]	Gets the data type identifier.
	Dimensions [▶ 528]	Gets the dimensions of an array type
	EnumValues [▶ 529]	Enumeration specification (if enum)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	HasArrayInfo [▶ 529]	Gets a value indicating whether this instance has dimension information.
	HasEnumInfo [▶ 530]	Gets a value indicating whether this instance has enum information.
	HasRpcMethods [▶ 530]	Gets a value indicating whether this instance has RPC methods (Struct types only)
	HasSubItemInfo [▶ 531]	Gets a value indicating whether this instance has sub items.
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsJaggedArray [▶ 531]	Gets a value indicating whether this dataType is a jagged array.
	IsOversamplingArray [▶ 532]	Gets a value indicating whether this instance is an oversampling array.
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)

	Name	Description
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	IsSubItem [▶ 532]	Determines whether this object is an SubItem (DataType Member, true) or a native DataType (false)
	ManagedType [▶ 533]	Managed Type of the (simple,primitive) type.
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	RpcMethods [▶ 533]	Gets the RPC method descriptions
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubItems [▶ 534]	Get the SubElements of this ITcAdsDataType [▶ 524].

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.1 ITcAdsDataType.BaseType Property

Gets the Base Type of the Type (if enum, alias, array)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ITcAdsDataType BaseType { get; }
```

VB

```
ReadOnly Property BaseType As ITcAdsDataType
    Get
```

Property Value

Type: ITcAdsDataType [▶ 524]
The type of the base.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.2 ITcAdsDataType.BaseTypeName Property

Gets the name of the base type (if enum, alias, array)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string BaseTypeName { get; }
```

VB

```
ReadOnly Property BaseTypeName As String
    Get
```

Property Value

Type: String
The name of the base type.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.3 ITcAdsDataType.DataTypeId Property

Gets the data type identifier.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsDatatypeId DataTypeId { get; }
```

VB

```
ReadOnly Property DataTypeId As AdsDatatypeId
    Get
```

Property Value

Type: AdsDatatypeId [▶ 299]
The data type identifier.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.4 ITcAdsDataType.Dimensions Property

Gets the dimensions of an array type

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyDimensionCollection Dimensions { get; }
```

VB

```
ReadOnly Property Dimensions As ReadOnlyDimensionCollection
    Get
```

Property Value

Type: ReadOnlyDimensionCollection [▶ 1731]
The array infos.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.5 ITcAdsDataType.EnumValues Property

Enumeration specification (if enum)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyEnumValueCollection EnumValues { get; }
```

VB

```
ReadOnly Property EnumValues As ReadOnlyEnumValueCollection
    Get
```

Property Value

Type: ReadOnlyEnumValueCollection [▶ 1736]
The enum infos.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.6 ITcAdsDataType.HasArrayInfo Property

Gets a value indicating whether this instance has dimension information.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool HasArrayInfo { get; }
```

VB

```
ReadOnly Property HasArrayInfo As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has dimension information; otherwise, false.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.7 ITcAdsDataType.HasEnumInfo Property

Gets a value indicating whether this instance has enum information.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool HasEnumInfo { get; }
```

VB

```
ReadOnly Property HasEnumInfo As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has enum information; otherwise, false.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.8 ITcAdsDataType.HasRpcMethods Property

Gets a value indicating whether this instance has RPC methods (Struct types only)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool HasRpcMethods { get; }
```

VB

```
ReadOnly Property HasRpcMethods As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has RPC methods; otherwise, false.

Remarks

The DataType (Structure) must be marked with the PlcAttribute 'TcRpcEnable' to enable RpcMethods, otherwise RpcMethods are not passed through to the ADS symbolic information.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.9 ITcAdsDataType.HasSubItemInfo Property

Gets a value indicating whether this instance has sub items.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool HasSubItemInfo { get; }
```

VB

```
ReadOnly Property HasSubItemInfo As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has sub items; otherwise, false.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.10 ITcAdsDataType.IsJaggedArray Property

Gets a value indicating whether this dataType is a jagged array.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsJaggedArray { get; }
```

VB

```
ReadOnly Property IsJaggedArray As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is jagged array; otherwise, false.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.11 ITcAdsDataType.IsOversamplingArray Property

Gets a value indicating whether this instance is an oversampling array.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsOversamplingArray { get; }
```

VB

```
ReadOnly Property IsOversamplingArray As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is oversampling array; otherwise, false.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.12 ITcAdsDataType.IsSubItem Property

Determines whether this object is an SubItem (DataType Member, true) or a native DataType (false)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsSubItem { get; }
```

VB

```
ReadOnly Property IsSubItem As Boolean
    Get
```

Return Value

Type: Boolean
true if [is sub item]; otherwise, false.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.13 ITcAdsDataType.ManagedType Property

Managed Type of the (simple,primitive) type.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Type ManagedType { get; }
```

VB

```
ReadOnly Property ManagedType As Type
    Get
```

Property Value

Type: Type
The managed type of the data type.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.14 ITcAdsDataType.RpcMethods Property

Gets the RPC method descriptions

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyRpcMethodCollection RpcMethods { get; }
```

VB

```
ReadOnly Property RpcMethods As ReadOnlyRpcMethodCollection
    Get
```

Property Value

Type: ReadOnlyRpcMethodCollection [▶ 1762]
A list of the rpc methods.

Remarks

The DataType (Structure) must be marked with the PlcAttribute 'TcRpcEnable' to enable RpcMethods, otherwise RpcMethods are not passed through to the ADS symbolic information.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.45.1.15 ITcAdsDataType.SubItems Property

Get the SubElements of this ITcAdsDataType [▶ 524].

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlySubItemCollection SubItems { get; }
```

VB

```
ReadOnly Property SubItems As ReadOnlySubItemCollection
    Get
```

Property Value

Type: ReadOnlySubItemCollection [▶ 1768]
The subItems.

Reference

ITcAdsDataType Interface [▶ 524]

TwinCAT.Ads Namespace [▶ 108]

5.2.46 ITcAdsRpcInvoke Interface

Interface ITcAdsRpcInvoke

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ITcAdsRpcInvoke
```

VB

```
Public Interface ITcAdsRpcInvoke
```

Methods

	Name	Description
	InvokeRpcMethod(String, Int32, .Object.) [▶ 537]	Invokes the specified RPC Method
	InvokeRpcMethod(String, String, .Object.) [▶ 538]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 539]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 540]	Invokes the specified RPC Method.
	TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 541]	Invokes the specified RPC Method
	TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 542]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 543]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 544]	Invokes the specified RPC Method.

Remarks

This interface is used to invoke ADS RPC Method calls. This can be done 'symbolic' via the Method name or the MethodID of the method on the specified symbol. To activate the RPC Access within the PLC environment, its dataType (Structure, FB) must be marked with the PlcAttribute 'TcRpcEnable' to enable RpcMethods.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.46.1 ITcAdsRpcInvoke Methods

Methods

	Name	Description
	InvokeRpcMethod(String, Int32, .Object.) [▶ 537]	Invokes the specified RPC Method
	InvokeRpcMethod(String, String, .Object.) [▶ 538]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 539]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 540]	Invokes the specified RPC Method.
	TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 541]	Invokes the specified RPC Method
	TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 542]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 543]	Invokes the specified RPC Method

	Name	Description
	TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 544]	Invokes the specified RPC Method.

Reference

ITcAdsRpcInvoke Interface [▶ 534]

TwinCAT.Ads Namespace [▶ 108]

5.2.46.1.1 ITcAdsRpcInvoke.InvokeRpcMethod Method

Overload List

	Name	Description
	InvokeRpcMethod(String, Int32, .Object.) [▶ 537]	Invokes the specified RPC Method
	InvokeRpcMethod(String, String, .Object.) [▶ 538]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 539]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 540]	Invokes the specified RPC Method.

Reference

ITcAdsRpcInvoke Interface [▶ 534]

TwinCAT.Ads Namespace [▶ 108]

ITcAdsRpcInvoke.InvokeRpcMethod Method (String, Int32, .Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object InvokeRpcMethod(
    string symbolPath,
    int methodId,
    Object[] parameters
)
```

VB

```
Function InvokeRpcMethod (
    symbolPath As String,
    methodId As Integer,
    parameters As Object()
) As Object
```

Parameters

symbolPath	Type: System.String The symbol path.
methodId	Type: System.Int32 The method identifier.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
The return value of the Method (as object).

Reference

ITcAdsRpcInvoke Interface [▶ 534]

InvokeRpcMethod Overload [▶ 537]

TwinCAT.Ads Namespace [▶ 108]

ITcAdsRpcInvoke.InvokeRpcMethod Method (String, String, .Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object InvokeRpcMethod(
    string symbolPath,
    string methodName,
    Object[] parameters
)
```

VB

```
Function InvokeRpcMethod (
    symbolPath As String,
    methodName As String,
    parameters As Object()
) As Object
```

Parameters

symbolPath	Type: System.String The symbol path.
methodName	Type: System.String The method name.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
The return value of the Method (as object).

Reference

ITcAdsRpcInvoke Interface [▶ 534]

InvokeRpcMethod Overload [▶ 537]

TwinCAT.Ads Namespace [▶ 108]

ITcAdsRpcInvoke.InvokeRpcMethod Method (ITcAdsSymbol, Int32, .Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object InvokeRpcMethod(
    ITcAdsSymbol symbol,
    int methodId,
    Object[] parameters
)
```

VB

```
Function InvokeRpcMethod (
    symbol As ITcAdsSymbol,
    methodId As Integer,
    parameters As Object()
) As Object
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
methodId	Type: System.Int32 The method identifier / Virtual Function table index.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
The return value of the Method (as object).

Reference

ITcAdsRpcInvoke Interface [▶ 534]

InvokeRpcMethod Overload [▶ 537]

TwinCAT.Ads Namespace [▶ 108]

ITcAdsRpcInvoke.InvokeRpcMethod Method (ITcAdsSymbol, String, .Object.)

Invokes the specified RPC Method.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object InvokeRpcMethod(
    ITcAdsSymbol symbol,
    string methodName,
    Object[] parameters
)
```

VB

```
Function InvokeRpcMethod (
    symbol As ITcAdsSymbol,
    methodName As String,
    parameters As Object()
) As Object
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
methodName	Type: System.String Name of the method.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
The return value of the Method (as object).

Reference

ITcAdsRpcInvoke Interface [▶ 534]

InvokeRpcMethod Overload [▶ 537]

TwinCAT.Ads Namespace [▶ 108]

5.2.46.1.2 ITcAdsRpcInvoke.TryInvokeRpcMethod Method

Overload List

	Name	Description
	TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 541]	Invokes the specified RPC Method
	TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 542]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 543]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 544]	Invokes the specified RPC Method.

Reference

ITcAdsRpcInvoke Interface [▶ 534]

TwinCAT.Ads Namespace [▶ 108]

ITcAdsRpcInvoke.TryInvokeRpcMethod Method (String, Int32, .Object., Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsErrorCode TryInvokeRpcMethod(
    string symbolPath,
    int methodId,
    Object[] parameters,
    out Object retValue
)
```

VB

```
Function TryInvokeRpcMethod (
    symbolPath As String,
    methodId As Integer,
    parameters As Object(),
    <OutAttribute> ByRef retValue As Object
) As AdsErrorCode
```

Parameters

symbolPath	Type: System.String The symbol path.
methodId	Type: System.Int32 The method identifier.
parameters	Type: .System.Object. The parameters.
retValue	Type: System.Object. The return value of the RPC method as object.

Return Value

Type: AdsErrorCode [▶ 305]
The ADS Error Code.

Reference

ITcAdsRpcInvoke Interface [▶ 534]

TryInvokeRpcMethod Overload [▶ 541]

TwinCAT.Ads Namespace [▶ 108]

ITcAdsRpcInvoke.TryInvokeRpcMethod Method (String, String, .Object., Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsErrorCode TryInvokeRpcMethod(
    string symbolPath,
    string methodName,
    Object[] parameters,
    out Object retValue
)
```

VB

```
Function TryInvokeRpcMethod (
    symbolPath As String,
    methodName As String,
    parameters As Object(),
    <OutAttribute> ByRef retValue As Object
) As AdsErrorCode
```

Parameters

symbolPath	Type: System.String The symbol path.
methodName	Type: System.String The method name.
parameters	Type: .System.Object. The parameters.
retValue	Type: System.Object. The return value of the RPC method as object.

Return Value

Type: AdsErrorCode [▶ 305]
The ADS Error Code.

Reference

ITcAdsRpcInvoke Interface [▶ 534]

TryInvokeRpcMethod Overload [▶ 541]

TwinCAT.Ads Namespace [▶ 108]

ITcAdsRpcInvoke.TryInvokeRpcMethod Method (ITcAdsSymbol, Int32, .Object., Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsErrorCode TryInvokeRpcMethod(
    ITcAdsSymbol symbol,
    int methodId,
    Object[] parameters,
    out Object retValue
)
```

VB

```
Function TryInvokeRpcMethod (
    symbol As ITcAdsSymbol,
    methodId As Integer,
    parameters As Object(),
    <OutAttribute> ByRef retValue As Object
) As AdsErrorCode
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
methodId	Type: System.Int32 The method identifier / Virtual Function table index.
parameters	Type: .System.Object. The parameters.
retValue	Type: System.Object. The return value of the RPC method as object.

Return Value

Type: AdsErrorCode [▶ 305]
The ADS Error Code.

Reference

ITcAdsRpcInvoke Interface [▶ 534]

TryInvokeRpcMethod Overload [▶ 541]

TwinCAT.Ads Namespace [▶ 108]

ITcAdsRpcInvoke.TryInvokeRpcMethod Method (ITcAdsSymbol, String, .Object., Object.)

Invokes the specified RPC Method.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsErrorCode TryInvokeRpcMethod(
    ITcAdsSymbol symbol,
    string methodName,
    Object[] parameters,
    out Object retValue
)
```

VB

```
Function TryInvokeRpcMethod (
    symbol As ITcAdsSymbol,
    methodName As String,
    parameters As Object(),
    <OutAttribute> ByRef retValue As Object
) As AdsErrorCode
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
methodName	Type: System.String Name of the method.
parameters	Type: .System.Object. The parameters.
retValue	Type: System.Object. The return value of the RPC method as object.

Return Value

Type: AdsErrorCode [▶ 305]
The ADS Error Code.

Reference

ITcAdsRpcInvoke Interface [▶ 534]

TryInvokeRpcMethod Overload [▶ 541]

TwinCAT.Ads Namespace [▶ 108]

5.2.47 ITcAdsSubItem Interface

Interface ITcAdsSubItem

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ITcAdsSubItem : ITcAdsDataType,
    IDataType, IBitSize
```

VB

```
Public Interface ITcAdsSubItem
    Inherits ITcAdsDataType, IDataType, IBitSize
```

The ITcAdsSubItem type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 527]	Gets the Base Type of the Type (if enum, alias, array) (Inherited from ITcAdsDataType [▶ 524].)
	BaseTypeName [▶ 527]	Gets the name of the base type (if enum, alias, array) (Inherited from ITcAdsDataType [▶ 524].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	DataTypeId [▶ 528]	Gets the data type identifier. (Inherited from ITcAdsDataType [▶ 524].)
	Dimensions [▶ 528]	Gets the dimensions of an array type (Inherited from ITcAdsDataType [▶ 524].)
	EnumValues [▶ 529]	Enumeration specification (if enum) (Inherited from ITcAdsDataType [▶ 524].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	HasArrayInfo [▶ 529]	Gets a value indicating whether this instance has dimension information. (Inherited from ITcAdsDataType [▶ 524].)
	HasEnumInfo [▶ 530]	Gets a value indicating whether this instance has enum information. (Inherited from ITcAdsDataType [▶ 524].)
	HasRpcMethods [▶ 530]	Gets a value indicating whether this instance has RPC methods (Struct types only) (Inherited from ITcAdsDataType [▶ 524].)
	HasSubItemInfo [▶ 531]	Gets a value indicating whether this instance has sub items. (Inherited from ITcAdsDataType [▶ 524].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)

	Name	Description
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsJaggedArray [▶ 531]	Gets a value indicating whether this dataType is a jagged array. (Inherited from ITcAdsDataType [▶ 524].)
	IsOversamplingArray [▶ 532]	Gets a value indicating whether this instance is an oversampling array. (Inherited from ITcAdsDataType [▶ 524].)
	IsPersistent [▶ 548]	Gets a value indicating whether this sub element is persistent.
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	IsSubItem [▶ 532]	Determines whether this object is an SubItem (DataType Member, true) or a native DataType (false) (Inherited from ITcAdsDataType [▶ 524].)
	ManagedType [▶ 533]	Managed Type of the (simple,primitive) type. (Inherited from ITcAdsDataType [▶ 524].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Offset [▶ 548]	Gets the offset of the SubItem (in Bytes or Bits)
	RpcMethods [▶ 533]	Gets the RPC method descriptions (Inherited from ITcAdsDataType [▶ 524].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubItemName [▶ 549]	Gets the name of the subitem / Sub element.
	SubItems [▶ 534]	Get the SubElements of this ITcAdsDataType [▶ 524]. (Inherited from ITcAdsDataType [▶ 524].)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.47.1 ITcAdsSubItem Properties

The ITcAdsSubItem [▶ 544] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 527]	Gets the Base Type of the Type (if enum, alias, array) (Inherited from ITcAdsDataType [▶ 524].)

	Name	Description
	BaseTypeName [▶ 527]	Gets the name of the base type (if enum, alias, array) (Inherited from ITcAdsDataType [▶ 524].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	DataTypeId [▶ 528]	Gets the data type identifier. (Inherited from ITcAdsDataType [▶ 524].)
	Dimensions [▶ 528]	Gets the dimensions of an array type (Inherited from ITcAdsDataType [▶ 524].)
	EnumValues [▶ 529]	Enumeration specification (if enum) (Inherited from ITcAdsDataType [▶ 524].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	HasArrayInfo [▶ 529]	Gets a value indicating whether this instance has dimension information. (Inherited from ITcAdsDataType [▶ 524].)
	HasEnumInfo [▶ 530]	Gets a value indicating whether this instance has enum information. (Inherited from ITcAdsDataType [▶ 524].)
	HasRpcMethods [▶ 530]	Gets a value indicating whether this instance has RPC methods (Struct types only) (Inherited from ITcAdsDataType [▶ 524].)
	HasSubItemInfo [▶ 531]	Gets a value indicating whether this instance has sub items. (Inherited from ITcAdsDataType [▶ 524].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsJaggedArray [▶ 531]	Gets a value indicating whether this dataType is a jagged array. (Inherited from ITcAdsDataType [▶ 524].)
	IsOversamplingArray [▶ 532]	Gets a value indicating whether this instance is an oversampling array. (Inherited from ITcAdsDataType [▶ 524].)
	IsPersistent [▶ 548]	Gets a value indicating whether this sub element is persistent.
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	IsSubItem [▶ 532]	Determines whether this object is an SubItem (DataType Member, true) or a native DataType (false) (Inherited from ITcAdsDataType [▶ 524].)

	Name	Description
	ManagedType [▶ 533]	Managed Type of the (simple,primitive) type. (Inherited from ITcAdsDataType [▶ 524].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Offset [▶ 548]	Gets the offset of the SubItem (in Bytes or Bits)
	RpcMethods [▶ 533]	Gets the RPC method descriptions (Inherited from ITcAdsDataType [▶ 524].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubItemName [▶ 549]	Gets the name of the subitem / Sub element.
	SubItems [▶ 534]	Get the SubElements of this ITcAdsDataType [▶ 524]. (Inherited from ITcAdsDataType [▶ 524].)

Reference

ITcAdsSubItem Interface [▶ 544]

TwinCAT.Ads Namespace [▶ 108]

5.2.47.1.1 ITcAdsSubItem.IsPersistent Property

Gets a value indicating whether this sub element is persistent.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsPersistent { get; }
```

VB

```
ReadOnly Property IsPersistent As Boolean
    Get
```

Property Value

Type: Boolean
true if this subelement is persistent; otherwise, false.

Reference

ITcAdsSubItem Interface [▶ 544]

TwinCAT.Ads Namespace [▶ 108]

5.2.47.1.2 ITcAdsSubItem.Offset Property

Gets the offset of the SubItem (in Bytes or Bits)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Offset { get; }
```

VB

```
ReadOnly Property Offset As Integer
    Get
```

Property Value

Type: Int32
The offset.

Remarks

If IsBitType [▶ 1516] indicates 'BitSize' then this value will be in Bits, otherwise Bytes.

Reference

ITcAdsSubItem Interface [▶ 544]

TwinCAT.Ads Namespace [▶ 108]

5.2.47.1.3 ITcAdsSubItem.SubItemName Property

Gets the name of the subitem / Sub element.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string SubItemName { get; }
```

VB

```
ReadOnly Property SubItemName As String
    Get
```

Property Value

Type: String
The name of the sub item.

Reference

ITcAdsSubItem Interface [▶ 544]

TwinCAT.Ads Namespace [▶ 108]

5.2.48 ITcAdsSymbol Interface

Defines an Interface for reading the ADS symbol information.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ITcAdsSymbol
```

VB

```
Public Interface ITcAdsSymbol
```

The ITcAdsSymbol type exposes the following members.

Properties

	Name	Description
	Comment [▶ 551]	Gets the comment behind the variable declaration.
	IndexGroup [▶ 551]	Gets the index group of the symbol.
	IndexOffset [▶ 552]	Gets the index offset of the symbol.
	Name [▶ 552]	Gets the name of the symbol.
	Size [▶ 553]	Gets the size of the symbol (in Bytes or bits dependent on IsBitType [▶ 555]).

Remarks

The most sophisticated implementation of this interface is ITcAdsSymbol"/>

Reference

TwinCAT.Ads Namespace [▶ 108]

TwinCAT.Ads.ITcAdsSymbol5 [▶ 566]

5.2.48.1 ITcAdsSymbol Properties

The ITcAdsSymbol [▶ 549] type exposes the following members.

Properties

	Name	Description
	Comment [▶ 551]	Gets the comment behind the variable declaration.
	IndexGroup [▶ 551]	Gets the index group of the symbol.
	IndexOffset [▶ 552]	Gets the index offset of the symbol.
	Name [▶ 552]	Gets the name of the symbol.
	Size [▶ 553]	Gets the size of the symbol (in Bytes or bits dependent on IsBitType [▶ 555]).

Reference

ITcAdsSymbol Interface [▶ 549]

TwinCAT.Ads Namespace [▶ 108]

5.2.48.1.1 ITcAdsSymbol.Comment Property

Gets the comment behind the variable declaration.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Comment { get; }
```

VB

```
ReadOnly Property Comment As String
    Get
```

Property Value

Type: String
Comment behind the variable declaration.

Reference

ITcAdsSymbol Interface [▶ 549]

TwinCAT.Ads Namespace [▶ 108]

5.2.48.1.2 ITcAdsSymbol.IndexGroup Property

Gets the index group of the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
long IndexGroup { get; }
```

VB

```
ReadOnly Property IndexGroup As Long
    Get
```

Property Value

Type: Int64
Index group of the symbol.

Reference

ITcAdsSymbol Interface [▶ 549]

TwinCAT.Ads Namespace [▶ 108]

5.2.48.1.3 ITcAdsSymbol.IndexOffset Property

Gets the index offset of the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
long IndexOffset { get; }
```

VB

```
ReadOnly Property IndexOffset As Long
    Get
```

Property Value

Type: Int64
Index offset of the symbol.

Reference

ITcAdsSymbol Interface [▶ 549]

TwinCAT.Ads Namespace [▶ 108]

5.2.48.1.4 ITcAdsSymbol.Name Property

Gets the name of the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Name { get; }
```

VB

```
ReadOnly Property Name As String
    Get
```

Property Value

Type: String
Name of the symbol.

Reference

ITcAdsSymbol Interface [▶ 549]

TwinCAT.Ads Namespace [▶ 108]

5.2.48.1.5 ITcAdsSymbol.Size Property

Gets the size of the symbol (in Bytes or bits dependent on IsBitType [▶ 555]).

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Size { get; }
```

VB

```
ReadOnly Property Size As Integer
    Get
```

Property Value

Type: Int32
Size of the symbol in Bytes / Bits

Reference

ITcAdsSymbol Interface [▶ 549]

TwinCAT.Ads Namespace [▶ 108]

5.2.49 ITcAdsSymbol2 Interface

Interface ITcAdsSymbol2 (extends ITcAdsSymbol [▶ 549])

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ITcAdsSymbol2 : ITcAdsSymbol
```

VB

```
Public Interface ITcAdsSymbol2
    Inherits ITcAdsSymbol
```

The ITcAdsSymbol2 type exposes the following members.

Properties

	Name	Description
	Comment [▶ 551]	Gets the comment behind the variable declaration. (Inherited from ITcAdsSymbol [▶ 549].)
	ContextMask [▶ 555]	Gets the ContextMask of the symbol, indicating the task the variable belongs to. If ContextMask is not zero use CyclicInContext [▶ 401] or OnChangeInContext [▶ 401] to add notifications.
	IndexGroup [▶ 551]	Gets the index group of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IndexOffset [▶ 552]	Gets the index offset of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)

	Name	Description
	IsBitType [▶ 555]	Indicates if the BitValue flag is set for the symbol.
	IsPersistent [▶ 556]	Indicates if the Persistent flag is set for the symbol.
	IsReadOnly [▶ 556]	Indicates if the ReadOnly flag is set for the symbol.
	IsTcComInterfacePointer [▶ 557]	Indicates if the TcComInterfacePointer flag is set for the symbol.
	IsTypeGuid [▶ 557]	Indicates if the TypeGuid flag is set for the symbol.
	Name [▶ 552]	Gets the name of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	Size [▶ 553]	Gets the size of the symbol (in Bytes or bits dependent on IsBitType [▶ 555]). (Inherited from ITcAdsSymbol [▶ 549].)

Remarks

The most sophisticated implementation of this interface is ITcAdsSymbol [▶ 549]"/>

Reference

TwinCAT.Ads Namespace [▶ 108]

TwinCAT.Ads.ITcAdsSymbol5 [▶ 566]

5.2.49.1 ITcAdsSymbol2 Properties

The ITcAdsSymbol2 [▶ 553] type exposes the following members.

Properties

	Name	Description
	Comment [▶ 551]	Gets the comment behind the variable declaration. (Inherited from ITcAdsSymbol [▶ 549].)
	ContextMask [▶ 555]	Gets the ContextMask of the symbol, indicating the task the variable belongs to. If ContextMask is not zero use CyclicInContext [▶ 401] or OnChangeInContext [▶ 401] to add notifications.
	IndexGroup [▶ 551]	Gets the index group of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IndexOffset [▶ 552]	Gets the index offset of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IsBitType [▶ 555]	Indicates if the BitValue flag is set for the symbol.
	IsPersistent [▶ 556]	Indicates if the Persistent flag is set for the symbol.
	IsReadOnly [▶ 556]	Indicates if the ReadOnly flag is set for the symbol.
	IsTcComInterfacePointer [▶ 557]	Indicates if the TcComInterfacePointer flag is set for the symbol.
	IsTypeGuid [▶ 557]	Indicates if the TypeGuid flag is set for the symbol.
	Name [▶ 552]	Gets the name of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)

	Name	Description
	Size [▶ 553]	Gets the size of the symbol (in Bytes or bits dependent on IsBitType [▶ 555]). (Inherited from ITcAdsSymbol [▶ 549].)

Reference

ITcAdsSymbol2 Interface [▶ 553]

TwinCAT.Ads Namespace [▶ 108]

5.2.49.1.1 ITcAdsSymbol2.ContextMask Property

Gets the ContextMask of the symbol, indicating the task the variable belongs to. If ContextMask is not zero use CyclicInContext [▶ 401] or OnChangeInContext [▶ 401] to add notifications.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int ContextMask { get; }
```

VB

```
ReadOnly Property ContextMask As Integer
    Get
```

Property Value

Type: Int32

Reference

ITcAdsSymbol2 Interface [▶ 553]

TwinCAT.Ads Namespace [▶ 108]

5.2.49.1.2 ITcAdsSymbol2.IsBitType Property

Indicates if the BitValue flag is set for the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsBitType { get; }
```

VB

```
ReadOnly Property IsBitType As Boolean
    Get
```

Property Value

Type: Boolean
true if is BitValue, otherwise false.

Reference

ITcAdsSymbol2 Interface [▶ 553]

TwinCAT.Ads Namespace [▶ 108]

5.2.49.1.3 ITcAdsSymbol2.IsPersistent Property

Indicates if the Persistent flag is set for the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsPersistent { get; }
```

VB

```
ReadOnly Property IsPersistent As Boolean
    Get
```

Property Value

Type: Boolean
true if persistent, otherwise false.

Reference

ITcAdsSymbol2 Interface [▶ 553]

TwinCAT.Ads Namespace [▶ 108]

5.2.49.1.4 ITcAdsSymbol2.IsReadOnly Property

Indicates if the ReadOnly flag is set for the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsReadOnly { get; }
```

VB

```
ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if ReadOnly, otherwise false.

Reference

ITcAdsSymbol2 Interface [▶ 553]

TwinCAT.Ads Namespace [▶ 108]

5.2.49.1.5 ITcAdsSymbol2.IsTcComInterfacePointer Property

Indicates if the TcComInterfacePointer flag is set for the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsTcComInterfacePointer { get; }
```

VB

```
ReadOnly Property IsTcComInterfacePointer As Boolean
    Get
```

Property Value

Type: Boolean
true if is TcComInterfacePointer, otherwise false.

Reference

ITcAdsSymbol2 Interface [▶ 553]

TwinCAT.Ads Namespace [▶ 108]

5.2.49.1.6 ITcAdsSymbol2.IsTypeGuid Property

Indicates if the TypeGuid flag is set for the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsTypeGuid { get; }
```

VB

```
ReadOnly Property IsTypeGuid As Boolean
    Get
```

Property Value

Type: Boolean
true if is TypeGuid, otherwise false.

Reference

ITcAdsSymbol2 Interface [▶ 553]

TwinCAT.Ads Namespace [▶ 108]

5.2.50 ITcAdsSymbol3 Interface

Interface ITcAdsSymbol3 (extends ITcAdsSymbol [▶ 549] ... ITcAdsSymbol3)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ITcAdsSymbol3 : ITcAdsSymbol2,
    ITcAdsSymbol
```

VB

```
Public Interface ITcAdsSymbol3
    Inherits ITcAdsSymbol2, ITcAdsSymbol
```

The ITcAdsSymbol3 type exposes the following members.

Properties

	Name	Description
	ArrayDimensions [▶ 560]	Gets the array dimensions.
	ArrayInfos [▶ 560]	Gets the collection of Array Infos.
	Comment [▶ 551]	Gets the comment behind the variable declaration. (Inherited from ITcAdsSymbol [▶ 549].)
	ContextMask [▶ 555]	Gets the ContextMask of the symbol, indicating the task the variable belongs to. If ContextMask is not zero use CyclicInContext [▶ 401] or OnChangeInContext [▶ 401] to add notifications. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IndexGroup [▶ 551]	Gets the index group of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IndexOffset [▶ 552]	Gets the index offset of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IsBitType [▶ 555]	Indicates if the BitValue flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsPersistent [▶ 556]	Indicates if the Persistent flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsReadOnly [▶ 556]	Indicates if the ReadOnly flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsTcComInterfacePointer [▶ 557]	Indicates if the TcComInterfacePointer flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)

	Name	Description
	IsTypeGuid [▶ 557]	Indicates if the TypeGuid flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	Name [▶ 552]	Gets the name of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	Size [▶ 553]	Gets the size of the symbol (in Bytes or bits dependent on IsBitType [▶ 555]). (Inherited from ITcAdsSymbol [▶ 549].)

Remarks

The most sophisticated implementation of this interface is ITcAdsSymbol [▶ 549]"/>

Reference

TwinCAT.Ads Namespace [▶ 108]

TwinCAT.Ads.ITcAdsSymbol5 [▶ 566]

5.2.50.1 ITcAdsSymbol3 Properties

The ITcAdsSymbol3 [▶ 558] type exposes the following members.

Properties

	Name	Description
	ArrayDimensions [▶ 560]	Gets the array dimensions.
	ArrayInfos [▶ 560]	Gets the collection of Array Infos.
	Comment [▶ 551]	Gets the comment behind the variable declaration. (Inherited from ITcAdsSymbol [▶ 549].)
	ContextMask [▶ 555]	Gets the ContextMask of the symbol, indicating the task the variable belongs to. If ContextMask is not zero use CyclicInContext [▶ 401] or OnChangeInContext [▶ 401] to add notifications. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IndexGroup [▶ 551]	Gets the index group of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IndexOffset [▶ 552]	Gets the index offset of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IsBitType [▶ 555]	Indicates if the BitValue flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsPersistent [▶ 556]	Indicates if the Persistent flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsReadOnly [▶ 556]	Indicates if the ReadOnly flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsTcComInterfacePointer [▶ 557]	Indicates if the TcComInterfacePointer flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsTypeGuid [▶ 557]	Indicates if the TypeGuid flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	Name [▶ 552]	Gets the name of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	Size [▶ 553]	Gets the size of the symbol (in Bytes or bits dependent on IsBitType [▶ 555]). (Inherited from ITcAdsSymbol [▶ 549].)

Reference

ITcAdsSymbol3 Interface [▶ 558]

TwinCAT.Ads Namespace [▶ 108]

5.2.50.1.1 ITcAdsSymbol3.ArrayDimensions Property

Gets the array dimensions.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int ArrayDimensions { get; }
```

VB

```
ReadOnly Property ArrayDimensions As Integer
    Get
```

Property Value

Type: Int32
The array dimensions.

Reference

ITcAdsSymbol3 Interface [▶ 558]

TwinCAT.Ads Namespace [▶ 108]

5.2.50.1.2 ITcAdsSymbol3.ArrayInfos Property

Gets the collection of Array Infos.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsDatatypeArrayInfo[] ArrayInfos { get; }
```

VB

```
ReadOnly Property ArrayInfos As AdsDatatypeArrayInfo()
    Get
```

Property Value

Type: .AdsDatatypeArrayInfo [▶ 296].
The array infos.

Reference

ITcAdsSymbol3 Interface [▶ 558]

TwinCAT.Ads Namespace [▶ 108]

5.2.51 ITcAdsSymbol4 Interface

Interface ITcAdsSymbol4 (extends ITcAdsSymbol [▶ 549] ... ITcAdsSymbol3 [▶ 558]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ITcAdsSymbol4 : ITcAdsSymbol3,
    ITcAdsSymbol2, ITcAdsSymbol
```

VB

```
Public Interface ITcAdsSymbol4
    Inherits ITcAdsSymbol3, ITcAdsSymbol2, ITcAdsSymbol
```

The ITcAdsSymbol4 type exposes the following members.

Properties

	Name	Description
	ArrayDimensions [▶ 560]	Gets the array dimensions. (Inherited from ITcAdsSymbol3 [▶ 558].)
	ArrayInfos [▶ 560]	Gets the collection of Array Infos. (Inherited from ITcAdsSymbol3 [▶ 558].)
	Attributes [▶ 563]	Gets the attribute count of the ITcAdsSymbol [▶ 549]
	BitSize [▶ 564]	Gets the Bit Size of the Symbol
	ByteSize [▶ 564]	Gets the Byte Size of the Symbol
	Category [▶ 565]	Gets the Data Type Category
	Comment [▶ 551]	Gets the comment behind the variable declaration. (Inherited from ITcAdsSymbol [▶ 549].)
	ContextMask [▶ 555]	Gets the ContextMask of the symbol, indicating the task the variable belongs to. If ContextMask is not zero use CyclicInContext [▶ 401] or OnChangeInContext [▶ 401] to add notifications. (Inherited from ITcAdsSymbol2 [▶ 553].)
	HasRpcMethods [▶ 565]	Gets a value indicating whether this instance has RPC methods (Struct types only)
	IndexGroup [▶ 551]	Gets the index group of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IndexOffset [▶ 552]	Gets the index offset of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IsBitType [▶ 555]	Indicates if the BitValue flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsPersistent [▶ 556]	Indicates if the Persistent flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsReadOnly [▶ 556]	Indicates if the ReadOnly flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)

	Name	Description
	IsTcComInterfacePointer [▶ 557]	Indicates if the TcComInterfacePointer flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsTypeGuid [▶ 557]	Indicates if the TypeGuid flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	Name [▶ 552]	Gets the name of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	RpcMethods [▶ 566]	Gets the RPC method descriptions
	Size [▶ 553]	Gets the size of the symbol (in Bytes or bits dependent on IsBitType [▶ 555]). (Inherited from ITcAdsSymbol [▶ 549].)

Remarks

The most sophisticated implementation of this interface is ITcAdsSymbol [▶ 549]"/>

Reference

TwinCAT.Ads Namespace [▶ 108]

TwinCAT.Ads.ITcAdsSymbol5 [▶ 566]

5.2.51.1 ITcAdsSymbol4 Properties

The ITcAdsSymbol4 [▶ 561] type exposes the following members.

Properties

	Name	Description
	ArrayDimensions [▶ 560]	Gets the array dimensions. (Inherited from ITcAdsSymbol3 [▶ 558].)
	ArrayInfos [▶ 560]	Gets the collection of Array Infos. (Inherited from ITcAdsSymbol3 [▶ 558].)
	Attributes [▶ 563]	Gets the attribute count of the ITcAdsSymbol [▶ 549]
	BitSize [▶ 564]	Gets the Bit Size of the Symbol
	ByteSize [▶ 564]	Gets the Byte Size of the Symbol
	Category [▶ 565]	Gets the Data Type Category
	Comment [▶ 551]	Gets the comment behind the variable declaration. (Inherited from ITcAdsSymbol [▶ 549].)
	ContextMask [▶ 555]	Gets the ContextMask of the symbol, indicating the task the variable belongs to. If ContextMask is not zero use CyclicInContext [▶ 401] or OnChangeInContext [▶ 401] to add notifications. (Inherited from ITcAdsSymbol2 [▶ 553].)
	HasRpcMethods [▶ 565]	Gets a value indicating whether this instance has RPC methods (Struct types only)
	IndexGroup [▶ 551]	Gets the index group of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IndexOffset [▶ 552]	Gets the index offset of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)

	Name	Description
	IsBitType [▶ 555]	Indicates if the BitValue flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsPersistent [▶ 556]	Indicates if the Persistent flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsReadOnly [▶ 556]	Indicates if the ReadOnly flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsTcComInterfacePointer [▶ 557]	Indicates if the TcComInterfacePointer flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsTypeGuid [▶ 557]	Indicates if the TypeGuid flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	Name [▶ 552]	Gets the name of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	RpcMethods [▶ 566]	Gets the RPC method descriptions
	Size [▶ 553]	Gets the size of the symbol (in Bytes or bits dependent on IsBitType [▶ 555]). (Inherited from ITcAdsSymbol [▶ 549].)

Reference

ITcAdsSymbol4 Interface [▶ 561]

TwinCAT.Ads Namespace [▶ 108]

5.2.51.1.1 ITcAdsSymbol4.Attributes Property

Gets the attribute count of the ITcAdsSymbol [▶ 549]

Gets the attributes of the ITcAdsSymbol [▶ 549]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyTypeAttributeCollection Attributes { get; }
```

VB

```
ReadOnly Property Attributes As ReadOnlyTypeAttributeCollection
    Get
```

Property Value

Type: ReadOnlyTypeAttributeCollection [▶ 1776]
The attribute count.

Property Value

Type: ReadOnlyTypeAttributeCollection [▶ 1776]
The attributes.

Reference

ITcAdsSymbol4 Interface [▶ 561]

TwinCAT.Ads Namespace [▶ 108]

5.2.51.1.2 ITcAdsSymbol4.BitSize Property

Gets the Bit Size of the Symbol

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int BitSize { get; }
```

VB

```
ReadOnly Property BitSize As Integer
    Get
```

Property Value

Type: Int32
The Bit Size of the underlying data type.

Reference

ITcAdsSymbol4 Interface [▶ 561]

TwinCAT.Ads Namespace [▶ 108]

5.2.51.1.3 ITcAdsSymbol4.ByteSize Property

Gets the Byte Size of the Symbol

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int ByteSize { get; }
```

VB

```
ReadOnly Property ByteSize As Integer
    Get
```

Property Value

Type: Int32
The Byte Size of the underlying data type.

Reference

ITcAdsSymbol4 Interface [▶ 561]

TwinCAT.Ads Namespace [▶ 108]

5.2.51.1.4 ITcAdsSymbol4.Category Property

Gets the Data Type Category

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
DataTypeCategory Category { get; }
```

VB

```
ReadOnly Property Category As DataTypeCategory
    Get
```

Property Value

Type: DataTypeCategory [▶ 1189]
The category.

Reference

ITcAdsSymbol4 Interface [▶ 561]

TwinCAT.Ads Namespace [▶ 108]

5.2.51.1.5 ITcAdsSymbol4.HasRpcMethods Property

Gets a value indicating whether this instance has RPC methods (Struct types only)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool HasRpcMethods { get; }
```

VB

```
ReadOnly Property HasRpcMethods As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has RPC methods; otherwise, false.

Remarks

The DataType (Structure) must be marked with the PlcAttribute 'TcRpcEnable' to enable RpcMethods, otherwise RpcMethods are not passed through to the ADS symbolic information.

Reference

ITcAdsSymbol4 Interface [▶ 561]

TwinCAT.Ads Namespace [▶ 108]

5.2.51.1.6 ITcAdsSymbol4.RpcMethods Property

Gets the RPC method descriptions

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyRpcMethodCollection RpcMethods { get; }
```

VB

```
ReadOnly Property RpcMethods As ReadOnlyRpcMethodCollection
    Get
```

Property Value

Type: ReadOnlyRpcMethodCollection [▶ 1762]
The RPC methods.

Remarks

The DataType (Structure) must be marked with the PlcAttribute 'TcRpcEnable' to enable RpcMethods, otherwise RpcMethods are not passed through to the ADS symbolic information.

Reference

ITcAdsSymbol4 Interface [▶ 561]

TwinCAT.Ads Namespace [▶ 108]

5.2.52 ITcAdsSymbol5 Interface

Interface ITcAdsSymbol5 (extends ITcAdsSymbol [▶ 549] ... ITcAdsSymbol4 [▶ 561]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ITcAdsSymbol5 : ITcAdsSymbol4,
    ITcAdsSymbol3, ITcAdsSymbol2, ITcAdsSymbol
```

VB

```
Public Interface ITcAdsSymbol5
    Inherits ITcAdsSymbol4, ITcAdsSymbol3, ITcAdsSymbol2, ITcAdsSymbol
```

The ITcAdsSymbol5 type exposes the following members.

Properties

	Name	Description
	ArrayDimensions [▶ 560]	Gets the array dimensions. (Inherited from ITcAdsSymbol3 [▶ 558].)
	ArrayInfos [▶ 560]	Gets the collection of Array Infos. (Inherited from ITcAdsSymbol3 [▶ 558].)

	Name	Description
	Attributes [▶ 563]	Gets the attribute count of the ITcAdsSymbol [▶ 549] (Inherited from ITcAdsSymbol4 [▶ 561].)
	BitSize [▶ 564]	Gets the Bit Size of the Symbol (Inherited from ITcAdsSymbol4 [▶ 561].)
	ByteSize [▶ 564]	Gets the Byte Size of the Symbol (Inherited from ITcAdsSymbol4 [▶ 561].)
	Category [▶ 565]	Gets the Data Type Category (Inherited from ITcAdsSymbol4 [▶ 561].)
	Comment [▶ 551]	Gets the comment behind the variable declaration. (Inherited from ITcAdsSymbol [▶ 549].)
	ContextMask [▶ 555]	Gets the ContextMask of the symbol, indicating the task the variable belongs to. If ContextMask is not zero use CyclicInContext [▶ 401] or OnChangeInContext [▶ 401] to add notifications. (Inherited from ITcAdsSymbol2 [▶ 553].)
	DataType [▶ 569]	Gets the Data type object of the Symbol.
	DataTypeId [▶ 569]	DataType identifier of the Symbol AdsDatatypeId [▶ 299]
	HasRpcMethods [▶ 565]	Gets a value indicating whether this instance has RPC methods (Struct types only) (Inherited from ITcAdsSymbol4 [▶ 561].)
	IndexGroup [▶ 551]	Gets the index group of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IndexOffset [▶ 552]	Gets the index offset of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IsBitType [▶ 555]	Indicates if the BitValue flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsPersistent [▶ 556]	Indicates if the Persistent flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsReadOnly [▶ 556]	Indicates if the ReadOnly flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsStatic [▶ 570]	Gets a value indicating whether the Symbol is static.
	IsTcComInterfacePointer [▶ 557]	Indicates if the TcComInterfacePointer flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsTypeGuid [▶ 557]	Indicates if the TypeGuid flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	Name [▶ 552]	Gets the name of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	RpcMethods [▶ 566]	Gets the RPC method descriptions (Inherited from ITcAdsSymbol4 [▶ 561].)
	Size [▶ 553]	Gets the size of the symbol (in Bytes or bits dependent on IsBitType [▶ 555]). (Inherited from ITcAdsSymbol [▶ 549].)
	TypeName [▶ 570]	Gets the name of the symbol data type.

Methods

	Name	Description
	IsRecursive [▶ 571]	Gets a value indicating whether this instance is recursive.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.52.1 ITcAdsSymbol5 Properties

The ITcAdsSymbol5 [▶ 566] type exposes the following members.

Properties

	Name	Description
	ArrayDimensions [▶ 560]	Gets the array dimensions. (Inherited from ITcAdsSymbol3 [▶ 558].)
	ArrayInfos [▶ 560]	Gets the collection of Array Infos. (Inherited from ITcAdsSymbol3 [▶ 558].)
	Attributes [▶ 563]	Gets the attribute count of the ITcAdsSymbol [▶ 549] (Inherited from ITcAdsSymbol4 [▶ 561].)
	BitSize [▶ 564]	Gets the Bit Size of the Symbol (Inherited from ITcAdsSymbol4 [▶ 561].)
	ByteSize [▶ 564]	Gets the Byte Size of the Symbol (Inherited from ITcAdsSymbol4 [▶ 561].)
	Category [▶ 565]	Gets the Data Type Category (Inherited from ITcAdsSymbol4 [▶ 561].)
	Comment [▶ 551]	Gets the comment behind the variable declaration. (Inherited from ITcAdsSymbol [▶ 549].)
	ContextMask [▶ 555]	Gets the ContextMask of the symbol, indicating the task the variable belongs to. If ContextMask is not zero use CyclicInContext [▶ 401] or OnChangeInContext [▶ 401] to add notifications. (Inherited from ITcAdsSymbol2 [▶ 553].)
	DataType [▶ 569]	Gets the Data type object of the Symbol.
	DataTypeId [▶ 569]	DataType identifier of the Symbol AdsDatatypeId [▶ 299]
	HasRpcMethods [▶ 565]	Gets a value indicating whether this instance has RPC methods (Struct types only) (Inherited from ITcAdsSymbol4 [▶ 561].)
	IndexGroup [▶ 551]	Gets the index group of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IndexOffset [▶ 552]	Gets the index offset of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	IsBitType [▶ 555]	Indicates if the BitValue flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsPersistent [▶ 556]	Indicates if the Persistent flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsReadOnly [▶ 556]	Indicates if the ReadOnly flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsStatic [▶ 570]	Gets a value indicating whether the Symbol is static.
	IsTcComInterfacePointer [▶ 557]	Indicates if the TcComInterfacePointer flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	IsTypeGuid [▶ 557]	Indicates if the TypeGuid flag is set for the symbol. (Inherited from ITcAdsSymbol2 [▶ 553].)
	Name [▶ 552]	Gets the name of the symbol. (Inherited from ITcAdsSymbol [▶ 549].)
	RpcMethods [▶ 566]	Gets the RPC method descriptions (Inherited from ITcAdsSymbol4 [▶ 561].)
	Size [▶ 553]	Gets the size of the symbol (in Bytes or bits dependent on IsBitType [▶ 555]). (Inherited from ITcAdsSymbol [▶ 549].)

	Name	Description
	TypeName [▶ 570]	Gets the name of the symbol data type.

Reference

ITcAdsSymbol5 Interface [▶ 566]

TwinCAT.Ads Namespace [▶ 108]

5.2.52.1.1 ITcAdsSymbol5.DataType Property

Gets the Data type object of the Symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ITcAdsDataType DataType { get; }
```

VB

```
ReadOnly Property DataType As ITcAdsDataType
    Get
```

Property Value

Type: ITcAdsDataType [▶ 524]
The data type object or NULL if not resolved.

Reference

ITcAdsSymbol5 Interface [▶ 566]

TwinCAT.Ads Namespace [▶ 108]

5.2.52.1.2 ITcAdsSymbol5.DataTypeId Property

DataType identifier of the Symbol AdsDatatypeId [▶ 299]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsDatatypeId DataTypeId { get; }
```

VB

```
ReadOnly Property DataTypeId As AdsDatatypeId
    Get
```

Property Value

Type: AdsDatatypeId [▶ 299]
Data type of the symbol.

Reference

ITcAdsSymbol5 Interface [▶ 566]

TwinCAT.Ads Namespace [▶ 108]

5.2.52.1.3 ITcAdsSymbol5.IsStatic Property

Gets a value indicating whether the Symbol is static.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsStatic { get; }
```

VB

```
ReadOnly Property IsStatic As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is static; otherwise, false.

Reference

ITcAdsSymbol5 Interface [▶ 566]

TwinCAT.Ads Namespace [▶ 108]

5.2.52.1.4 ITcAdsSymbol5.TypeName Property

Gets the name of the symbol data type.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string TypeName { get; }
```

VB

```
ReadOnly Property TypeName As String
    Get
```

Property Value

Type: String
Name of the symbol data type.

Reference

ITcAdsSymbol5 Interface [▶ 566]

TwinCAT.Ads Namespace [▶ 108]

5.2.52.2 ITcAdsSymbol5 Methods

The ITcAdsSymbol5 [▶ 566] type exposes the following members.

Methods

	Name	Description
	IsRecursive [▶ 571]	Gets a value indicating whether this instance is recursive.

Reference

ITcAdsSymbol5 Interface [▶ 566]

TwinCAT.Ads Namespace [▶ 108]

5.2.52.2.1 ITcAdsSymbol5.IsRecursive Method

Gets a value indicating whether this instance is recursive.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsRecursive(
    IEnumerable<ITcAdsSymbol5> parents
)
```

VB

```
Function IsRecursive (
    parents As IEnumerable(Of ITcAdsSymbol5)
) As Boolean
```

Parameters

parents	Type: System.Collections.Generic.IEnumerable.ITcAdsSymbol5 [▶ 566]. The parents.

Field Value

Type: Boolean
true if this instance is recursive; otherwise, false.

Return Value

Type: Boolean
true if the specified parents is recursive; otherwise, false.

Reference

ITcAdsSymbol5 Interface [▶ 566]

TwinCAT.Ads Namespace [▶ 108]

5.2.53 ITcAdsSymbolBrowser Interface

Interface ITcAdsSubSymbolProvider

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ITcAdsSymbolBrowser
```

VB

```
Public Interface ITcAdsSymbolBrowser
```

The ITcAdsSymbolBrowser type exposes the following members.

Properties

	Name	Description
	SubSymbols [▶ 572]	GetSubSymbols of this symbol.

Remarks

Interface used for ITcAdsSymbol [▶ 549] objects that support subsymbols (e.g. Arrays, References, Structs)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.53.1 ITcAdsSymbolBrowser Properties

The ITcAdsSymbolBrowser [▶ 572] type exposes the following members.

Properties

	Name	Description
	SubSymbols [▶ 572]	GetSubSymbols of this symbol.

Reference

ITcAdsSymbolBrowser Interface [▶ 572]

TwinCAT.Ads Namespace [▶ 108]

5.2.53.1.1 ITcAdsSymbolBrowser.SubSymbols Property

GetSubSymbols of this symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
TcAdsSymbolInfoCollection SubSymbols { get; }
```

VB

```
ReadOnly Property SubSymbols As TcAdsSymbolInfoCollection
    Get
```

Property Value

Type: TcAdsSymbolInfoCollection [▶ 805]
The SubSymbols.

Reference

ITcAdsSymbolBrowser Interface [▶ 572]

TwinCAT.Ads Namespace [▶ 108]

5.2.54 NotificationSettings Class

Notification communication settings

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.NotificationSettings
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class NotificationSettings : INotificationSettings,
    IComparable<INotificationSettings>
```

VB

```
Public Class NotificationSettings
    Implements INotificationSettings, IComparable(Of INotificationSettings)
```

The NotificationSettings type exposes the following members.

Constructors

	Name	Description
	NotificationSettings [▶ 574]	Initializes a new instance of the NotificationSettings class.

Properties

	Name	Description
	CycleTime [▶ 575]	Gets or sets the cycle time (in milliseconds) for AdsNotifications.
	Default [▶ 576]	Gets the default Settings (AdsTransMode.OnChange, CycleTime 200 ms, MaxDelay: off)

	Name	Description
	MaxDelay [▶ 576]	Gets or sets the Maximum Delay Time (in milliseconds) for AdsNotifications.
	NotificationMode [▶ 577]	Gets or sets the ADS Transmission mode.

Methods

	Name	Description
	CompareTo [▶ 578]	Compares this NotificationSettings in term of priorities to the other NotificationSettings.
	Equals [▶ 579]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 579]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Operators

	Name	Description
	Equality [▶ 580]	Operator==
	Inequality [▶ 581]	Implements the != operator.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.54.1 NotificationSettings Constructor

Initializes a new instance of the NotificationSettings [▶ 573] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public NotificationSettings(
    AdsTransMode mode,
    int cycleTime,
    int maxDelay
)
```

VB

```
Public Sub New (
    mode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer
)
```

Parameters

mode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] The ADS Transmission mode.
cycleTime	Type: System.Int32 The cycle time in ms.
maxDelay	Type: System.Int32 The maximum delay in ms

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

5.2.54.2 NotificationSettings Properties

The NotificationSettings [▶ 573] type exposes the following members.

Properties

	Name	Description
	CycleTime [▶ 575]	Gets or sets the cycle time (in milliseconds) for AdsNotifications.
S	Default [▶ 576]	Gets the default Settings (AdsTransMode.OnChange, CycleTime 200 ms, MaxDelay: off)
	MaxDelay [▶ 576]	Gets or sets the Maximum Delay Time (in milliseconds) for AdsNotifications.
	NotificationMode [▶ 577]	Gets or sets the ADS Transmission mode.

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

5.2.54.2.1 NotificationSettings.CycleTime Property

Gets or sets the cycle time (in milliseconds) for AdsNotifications.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int CycleTime { get; }
```

VB

```
Public ReadOnly Property CycleTime As Integer
    Get
```

Property Value

Type: Int32
The cycle time.

Remarks

The ADS server checks if the value changes in this time slice. The unit is 1ms

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

5.2.54.2.2 NotificationSettings.Default Property

Gets the default Settings (AdsTransMode.OnChange, CycleTime 200 ms, MaxDelay: off)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static NotificationSettings Default { get; }
```

VB

```
Public Shared ReadOnly Property Default As NotificationSettings
    Get
```

Property Value

Type: NotificationSettings [▶ 573]
The default.

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

5.2.54.2.3 NotificationSettings.MaxDelay Property

Gets or sets the Maximum Delay Time (in milliseconds) for AdsNotifications.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int MaxDelay { get; }
```

VB

```
Public ReadOnly Property MaxDelay As Integer
    Get
```

Property Value

Type: Int32
The maximum Delay time for ADS Notifications.

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

5.2.54.2.4 NotificationSettings.NotificationMode Property

Gets or sets the ADS Transmission mode.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsTransMode NotificationMode { get; }
```

VB

```
Public ReadOnly Property NotificationMode As AdsTransMode
    Get
```

Property Value

Type: AdsTransMode [▶ 401]
The Transmission mode.

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

5.2.54.3 NotificationSettings Methods

The NotificationSettings [▶ 573] type exposes the following members.

Methods

	Name	Description
	CompareTo [▶ 578]	Compares this NotificationSettings [▶ 573] in term of priorities to the other NotificationSettings [▶ 573].
	Equals [▶ 579]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	GetHashCode [▶ 579]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

5.2.54.3.1 NotificationSettings.CompareTo Method

Compares this NotificationSettings [▶ 573] in term of priorities to the other NotificationSettings [▶ 573].

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int CompareTo(
    INotificationSettings other
)
```

VB

```
Public Function CompareTo (
    other As INotificationSettings
) As Integer
```

Parameters

other	Type: TwinCAT.TypeSystem.INotificationSettings [▶ 1566] The other.

Return Value

Type: Int32
1: this has higher priority (shorter times), 0: Equal, 1: Lower prioritiy

Implements

IComparable.T..CompareTo(T)

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

NotificationSettingsPriorityComparer

5.2.54.3.2 NotificationSettings.Equals Method

Equals

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool Equals(
    Object obj
)
```

VB

```
Public Overrides Function Equals (
    obj As Object
) As Boolean
```

Parameters

obj	Type: System.Object The object to compare with the current object.

Return Value

Type: Boolean
true if the specified Object is equal to this instance; otherwise, false.

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

5.2.54.3.3 NotificationSettings.GetHashCode Method

Gets the HashCode of the Address

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override int GetHashCode()
```

VB

```
Public Overrides Function GetHashCode As Integer
```

Return Value

Type: Int32
A hash code for this instance, suitable for use in hashing algorithms and data structures like a hash table.

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

5.2.54.4 NotificationSettings Operators

The NotificationSettings [▶ 573] type exposes the following members.

Operators

	Name	Description
	Equality [▶ 580]	Operator==
	Inequality [▶ 581]	Implements the != operator.

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

5.2.54.4.1 NotificationSettings.Equality Operator

Operator==

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator ==(
    NotificationSettings o1,
    NotificationSettings o2
)
```

VB

```
Public Shared Operator = (
    o1 As NotificationSettings,
    o2 As NotificationSettings
) As Boolean
```

Parameters

o1	Type: TwinCAT.Ads.NotificationSettings [▶ 573] The o1.
o2	Type: TwinCAT.Ads.NotificationSettings [▶ 573] The o2.

Return Value

Type: Boolean
The result of the operator.

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

5.2.54.4.2 NotificationSettings.Inequality Operator

Implements the != operator.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator !=(
    NotificationSettings o1,
    NotificationSettings o2
)
```

VB

```
Public Shared Operator <> (
    o1 As NotificationSettings,
    o2 As NotificationSettings
) As Boolean
```

Parameters

o1	Type: TwinCAT.Ads.NotificationSettings [▶ 573] The o1.
o2	Type: TwinCAT.Ads.NotificationSettings [▶ 573] The o2.

Return Value

Type: Boolean
The result of the operator.

Reference

NotificationSettings Class [▶ 573]

TwinCAT.Ads Namespace [▶ 108]

5.2.55 ReadOnlyTcAdsDataTypeCollection Class

Read only collection of ITcAdsDataType [▶ 524]" types.

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.ITcAdsDataType [▶ 524].
    TwinCAT.Ads.ReadOnlyTcAdsDataTypeCollection
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyTcAdsDataTypeCollection : ReadOnlyCollection<ITcAdsDataType>,
    IEnumerable<IDataType>, IEnumerable
```

VB

```
Public Class ReadOnlyTcAdsDataTypeCollection
    Inherits ReadOnlyCollection(Of ITcAdsDataType)
    Implements IEnumerable(Of IDataType), IEnumerable
```

The ReadOnlyTcAdsDataTypeCollection type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	Item.String. [▶ 583]	Gets the ITcAdsDataType [▶ 524] with the specified type name.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)

Methods

	Name	Description
	Contains(String) [▶ 585]	Determines whether this ReadOnlyTcAdsDataTypeCollection contains the specified type name.
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetDataType [▶ 586]	Tries to get the specified type name.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.55.1 ReadOnlyTcAdsDataTypeCollection Properties

The ReadOnlyTcAdsDataTypeCollection [▶ 581] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	Item.String. [▶ 583]	Gets the ITcAdsDataType [▶ 524] with the specified type name.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)

Reference

ReadOnlyTcAdsDataTypeCollection Class [▶ 581]

TwinCAT.Ads Namespace [▶ 108]

5.2.55.1.1 ReadOnlyTcAdsDataTypeCollection.Item Property

Overload List

	Name	Description
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	Item.String. [▶ 583]	Gets the ITcAdsDataType [▶ 524] with the specified type name.

Reference

ReadOnlyTcAdsDataTypeCollection Class [▶ 581]

TwinCAT.Ads Namespace [▶ 108]

ReadOnlyTcAdsDataTypeCollection.Item Property (String)

Gets the ITcAdsDataType [▶ 524] with the specified type name.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ITcAdsDataType this[
    string typeName
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    typeName As String
) As ITcAdsDataType
    Get
```

Parameters

typeName	Type: System.String Name of the type.

Return Value

Type: ITcAdsDataType [▶ 524]
ITcAdsDataType.

Reference

ReadOnlyTcAdsDataTypeCollection Class [▶ 581]

Item Overload [▶ 583]

TwinCAT.Ads Namespace [▶ 108]

5.2.55.2 ReadOnlyTcAdsDataTypeCollection Methods

The ReadOnlyTcAdsDataTypeCollection [▶ 581] type exposes the following members.

Methods

	Name	Description
	Contains(String) [▶ 585]	Determines whether this ReadOnlyTcAdsDataTypeCollection [▶ 581] contains the specified type name.
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

	Name	Description
	TryGetDataType [▶ 586]	Tries to get the specified type name.

Reference

ReadOnlyTcAdsDataTypeCollection Class [▶ 581]

TwinCAT.Ads Namespace [▶ 108]

5.2.55.2.1 ReadOnlyTcAdsDataTypeCollection.Contains Method

Overload List

	Name	Description
	Contains(String) [▶ 585]	Determines whether this ReadOnlyTcAdsDataTypeCollection [▶ 581] contains the specified type name.
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsDataType [▶ 524]..)

Reference

ReadOnlyTcAdsDataTypeCollection Class [▶ 581]

TwinCAT.Ads Namespace [▶ 108]

ReadOnlyTcAdsDataTypeCollection.Contains Method (String)

Determines whether this ReadOnlyTcAdsDataTypeCollection [▶ 581] contains the specified type name.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    string typeName
)
```

VB

```
Public Function Contains (
    typeName As String
) As Boolean
```

Parameters

typeName	Type: System.String Name of the type.

Return Value

Type: Boolean
true if the specified name is contained; otherwise, false.

Reference

ReadOnlyTcAdsDataTypeCollection Class [▶ 581]

Contains Overload [▶ 585]

TwinCAT.Ads Namespace [▶ 108]

5.2.55.2.2 ReadOnlyTcAdsDataTypeCollection.TryGetDataType Method

Tries to get the specified type name.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetDataType(
    string typeName,
    out ITcAdsDataType type
)
```

VB

```
Public Function TryGetDataType (
    typeName As String,
    <OutAttribute> ByRef type As ITcAdsDataType
) As Boolean
```

Parameters

typeName	Type: System.String Name of the type.
type	Type: TwinCAT.Ads.ITcAdsDataType [▶ 524]. The type.

Return Value

Type: Boolean
true if found, false otherwise.

Reference

ReadOnlyTcAdsDataTypeCollection Class [▶ 581]

TwinCAT.Ads Namespace [▶ 108]

5.2.56 RpcMethodNotSupportedException Class

Symbol Exception

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.Ads.AdsSymbolException [▶ 389]

TwinCAT.Ads.RpcMethodNotSupportedException

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class RpcMethodNotSupportedException : AdsSymbolException
```

VB

```
<SerializableAttribute>
Public Class RpcMethodNotSupportedException
    Inherits AdsSymbolException
```

The RpcMethodNotSupportedException type exposes the following members.

Constructors

	Name	Description
	RpcMethodNotSupportedException(Int32, ITcAdsSymbol) [▶ 589]	Initializes a new instance of the RpcMethodNotSupportedException class.
	RpcMethodNotSupportedException(String, ITcAdsSymbol) [▶ 589]	Initializes a new instance of the RpcMethodNotSupportedException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Fields

	Name	Description
	Symbol [▶ 397]	The symbol (Inherited from AdsSymbolException [▶ 389].)
	SymbolName [▶ 397]	The symbol (Inherited from AdsSymbolException [▶ 389].)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.56.1 RpcMethodNotSupportedException Constructor

Overload List

	Name	Description
	RpcMethodNotSupportedException(Int32, ITcAdsSymbol) [▶ 589]	Initializes a new instance of the RpcMethodNotSupportedException [▶ 586] class.
	RpcMethodNotSupportedException(String, ITcAdsSymbol) [▶ 589]	Initializes a new instance of the RpcMethodNotSupportedException [▶ 586] class.

Reference

RpcMethodNotSupportedException Class [▶ 586]

TwinCAT.Ads Namespace [▶ 108]

5.2.56.1.1 RpcMethodNotSupportedException Constructor (Int32, ITcAdsSymbol)

Initializes a new instance of the RpcMethodNotSupportedException [▶ 586] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public RpcMethodNotSupportedException(
    int vTableIndex,
    ITcAdsSymbol symbol
)
```

VB

```
Public Sub New (
    vTableIndex As Integer,
    symbol As ITcAdsSymbol
)
```

Parameters

vTableIndex	Type: System.Int32 Index of the v table.
symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.

Reference

RpcMethodNotSupportedException Class [▶ 586]

RpcMethodNotSupportedException Overload [▶ 588]

TwinCAT.Ads Namespace [▶ 108]

5.2.56.1.2 RpcMethodNotSupportedException Constructor (String, ITcAdsSymbol)

Initializes a new instance of the RpcMethodNotSupportedException [▶ 586] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public RpcMethodNotSupportedException(
    string methodName,
    ITcAdsSymbol symbol
)
```

VB

```
Public Sub New (
    methodName As String,
    symbol As ITcAdsSymbol
)
```

Parameters

methodName	Type: System.String Name of the method.
symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.

Reference

RpcMethodNotSupportedException Class [▶ 586]

RpcMethodNotSupportedException Overload [▶ 588]

TwinCAT.Ads Namespace [▶ 108]

5.2.56.2 RpcMethodNotSupportedException Properties

The RpcMethodNotSupportedException [▶ 586] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

RpcMethodNotSupportedException Class [▶ 586]

TwinCAT.Ads Namespace [▶ 108]

5.2.56.3 RpcMethodNotSupportedException Methods

The RpcMethodNotSupportedException [▶ 586] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

RpcMethodNotSupportedException Class [▶ 586]

TwinCAT.Ads Namespace [▶ 108]

5.2.56.4 RpcMethodNotSupportedException Events

The RpcMethodNotSupportedException [▶ 586] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

RpcMethodNotSupportedException Class [▶ 586]

TwinCAT.Ads Namespace [▶ 108]

5.2.56.5 RpcMethodNotSupportedException Fields

The RpcMethodNotSupportedException [▶ 586] type exposes the following members.

Fields

	Name	Description
	Symbol [▶ 397]	The symbol (Inherited from AdsSymbolException [▶ 389].)
	SymbolName [▶ 397]	The symbol (Inherited from AdsSymbolException [▶ 389].)

Reference

RpcMethodNotSupportedException Class [▶ 586]

TwinCAT.Ads Namespace [▶ 108]

5.2.57 SessionSettings Class

Session settings class

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.SessionSettings
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SessionSettings : IAdsSessionSettings,
    ISessionSettings
```

VB

```
Public Class SessionSettings
    Implements IAdsSessionSettings, ISessionSettings
```

The SessionSettings type exposes the following members.

Properties

	Name	Description
	Default [▶ 593]	Gets the default Settings (Synchronized).
	FastWriteThrough [▶ 594]	Gets a Settings object that configures the AdsSession for FastWriteThrough
	ResurrectionTime [▶ 595]	Gets or sets the resurrection time (Default: DefaultResurrectionTime [▶ 598])
	SymbolLoader [▶ 596]	Gets or sets the symbol loader settings
	Timeout [▶ 596]	Gets the ADS timeout in milliseconds.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)

	Name	Description
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Fields

	Name	Description
	DefaultCommunicationTimeout [▶ 598]	The default communication timeout (5 Seconds)
	DefaultResurrectionTime [▶ 598]	The default resurrection time (21 Seconds)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.57.1 SessionSettings Properties

The SessionSettings [▶ 592] type exposes the following members.

Properties

	Name	Description
	Default [▶ 593]	Gets the default Settings (Synchronized).
	FastWriteThrough [▶ 594]	Gets a Settings object that configures the AdsSession for FastWriteThrough
	ResurrectionTime [▶ 595]	Gets or sets the resurrection time (Default: DefaultResurrectionTime [▶ 598])
	SymbolLoader [▶ 596]	Gets or sets the symbol loader settings
	Timeout [▶ 596]	Gets the ADS timeout in milliseconds.

Reference

SessionSettings Class [▶ 592]

TwinCAT.Ads Namespace [▶ 108]

5.2.57.1.1 SessionSettings.Default Property

Gets the default Settings (Synchronized).

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static SessionSettings Default { get; }
```

VB

```
Public Shared ReadOnly Property Default As SessionSettings
    Get
```

Property Value

Type: SessionSettings [▶ 592]
The default settings.

Remarks

The following defaults are set here:

Setting	Description
Communication Timeout (Timeout [▶ 596])	Default communication timeout (DefaultCommunicationTimeout [▶ 598], default 5s)
Resurrection Timeout (ResurrectionTime [▶ 595])	Default communication timeout (DefaultResurrectionTime [▶ 598], default 21s)
Dynamic SymbolLoader settings SymbolLoader [▶ 596]	Synchronized mode activated (DefaultDynamic [▶ 622])

Reference

SessionSettings Class [▶ 592]

TwinCAT.Ads Namespace [▶ 108]

5.2.57.1.2 SessionSettings.FastWriteThrough Property

Gets a Settings object that configures the AdsSession for FastWriteThrough

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static SessionSettings FastWriteThrough { get; }
```

VB

```
Public Shared ReadOnly Property FastWriteThrough As SessionSettings
    Get
```

Property Value

Type: SessionSettings [▶ 592]
Session settings for a fast write through (with 200 ms Timeout).

Remarks

The settings typically can be used for polling clients, where the "FailFast" feature will be bypassed. That means, that communication fails doesn't trigger the FailFast interceptor and every Request will go out via ADS. This has the Drawback that communication Timeouts are longer and subsequent timeouts block the ADS mailbox (with the danger of overflows). So use this setting with care for specific purposes and should not be used for standard communication.

- No Resurrection time and therefore:
- No FailFastHandler [▶ 523] active.
- Default communication timeout 200ms.
- Not synchronized Notifications.

Reference

SessionSettings Class [▶ 592]

TwinCAT.Ads Namespace [▶ 108]

5.2.57.1.3 SessionSettings.ResurrectionTime Property

Gets or sets the resurrection time (Default: DefaultResurrectionTime [▶ 598])

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TimeSpan ResurrectionTime { get; set; }
```

VB

```
Public Property ResurrectionTime As TimeSpan
    Get
    Set
```

Property Value

Type: TimeSpan
The resurrection time.

Implements

IAdsSessionSettings.ResurrectionTime [▶ 522]

Remarks

The resurrection time is the time after a lost connection Lost [▶ 39] can be 'resurrected'. This time is set to 21 Seconds by default (a value greater than the standard Ethernet connection timeout of 20s). The reason for this timeout is not to flood the ADS mailbox with requests that cannot be handled by the ethernet infrastructure. As long this Timespan is not expired after a recognized Lost [▶ 39], no further data communication is done, and requests are immediately ('FailFast') answered by communication exceptions. **Change this value only for edge cases.**

Reference

SessionSettings Class [▶ 592]

TwinCAT.Ads Namespace [▶ 108]

5.2.57.1.4 SessionSettings.SymbolLoader Property

Gets or sets the symbol loader settings

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolLoaderSettings SymbolLoader { get; set; }
```

VB

```
Public Property SymbolLoader As SymbolLoaderSettings
    Get
    Set
```

Property Value

Type: SymbolLoaderSettings [▶ 615]
The symbol loader.

Implements

IAdsSessionSettings.SymbolLoader [▶ 522]

Reference

SessionSettings Class [▶ 592]

TwinCAT.Ads Namespace [▶ 108]

5.2.57.1.5 SessionSettings.Timeout Property

Gets the ADS timeout in milliseconds.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Timeout { get; }
```

VB

```
Public ReadOnly Property Timeout As Integer
    Get
```

Property Value

Type: Int32
The timeout.

Implements

IAdsSessionSettings.Timeout [▶ 523]

Reference

SessionSettings Class [▶ 592]

TwinCAT.Ads Namespace [▶ 108]

5.2.57.2 SessionSettings Methods

The SessionSettings [▶ 592] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

SessionSettings Class [▶ 592]

TwinCAT.Ads Namespace [▶ 108]

5.2.57.3 SessionSettings Fields

The SessionSettings [▶ 592] type exposes the following members.

Fields

	Name	Description
	DefaultCommunicationTimeout [▶ 598]	The default communication timeout (5 Seconds)
	DefaultResurrectionTime [▶ 598]	The default resurrection time (21 Seconds)

Reference

SessionSettings Class [▶ 592]

TwinCAT.Ads Namespace [▶ 108]

5.2.57.3.1 SessionSettings.DefaultCommunicationTimeout Field

The default communication timeout (5 Seconds)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static TimeSpan DefaultCommunicationTimeout
```

VB

```
Public Shared DefaultCommunicationTimeout As TimeSpan
```

Field Value

Type: TimeSpan

Reference

SessionSettings Class [▶ 592]

TwinCAT.Ads Namespace [▶ 108]

5.2.57.3.2 SessionSettings.DefaultResurrectionTime Field

The default resurrection time (21 Seconds)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static TimeSpan DefaultResurrectionTime
```

VB

```
Public Shared DefaultResurrectionTime As TimeSpan
```

Field Value

Type: TimeSpan

Reference

SessionSettings Class [▶ 592]

TwinCAT.Ads Namespace [▶ 108]

5.2.58 StateInfo Structure

The structure contains the ADS state and device state.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public struct StateInfo
```

VB

```
Public Structure StateInfo
```

The StateInfo type exposes the following members.

Constructors

	Name	Description
	StateInfo [▶ 599]	Initializes a new Instance of the StateInfo struct.

Properties

	Name	Description
	AdsState [▶ 600]	Gets or sets the ADS state of this StateInfo object.
	DeviceState [▶ 601]	Gets or sets the device state of this StateInfo object.

Methods

	Name	Description
	Equals(Object) [▶ 602]	Determines whether the specified Object is equal to this instance. (Overrides ValueType.Equals(Object).)
	Equals(StateInfo) [▶ 603]	Determines whether the specified StateInfo is equal to this instance.
	GetHashCode [▶ 603]	Returns a hash code for this instance. (Overrides ValueType.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns the fully qualified type name of this instance. (Inherited from ValueType.)

Operators

	Name	Description
	Equality [▶ 604]	Implements the ==.
	Inequality [▶ 605]	Implements the !=.

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.58.1 StateInfo Constructor

Initializes a new Instance of the StateInfo struct.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public StateInfo(
    AdsState adsState,
    short deviceState
)
```

VB

```
Public Sub New (
    adsState As AdsState,
    deviceState As Short
)
```

Parameters

adsState	Type: TwinCAT.Ads.AdsState [▶ 365] Ads state.
deviceState	Type: System.Int16 Device state.

Reference

StateInfo Structure [▶ 598]

TwinCAT.Ads Namespace [▶ 108]

5.2.58.2 StateInfo Properties

The StateInfo [▶ 598] type exposes the following members.

Properties

	Name	Description
	AdsState [▶ 600]	Gets or sets the ADS state of this StateInfo [▶ 598] object.
	DeviceState [▶ 601]	Gets or sets the device state of this StateInfo [▶ 598] object.

Reference

StateInfo Structure [▶ 598]

TwinCAT.Ads Namespace [▶ 108]

5.2.58.2.1 StateInfo.AdsState Property

Gets or sets the ADS state of this StateInfo [▶ 598] object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsState AdsState { get; set; }
```

VB

```
Public Property AdsState As AdsState
    Get
    Set
```

Property Value

Type: AdsState [▶ 365]

Reference

StateInfo Structure [▶ 598]

TwinCAT.Ads Namespace [▶ 108]

5.2.58.2.2 StateInfo.DeviceState Property

Gets or sets the device state of this StateInfo [▶ 598] object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public short DeviceState { get; set; }
```

VB

```
Public Property DeviceState As Short
    Get
    Set
```

Property Value

Type: Int16

Reference

StateInfo Structure [▶ 598]

TwinCAT.Ads Namespace [▶ 108]

5.2.58.3 StateInfo Methods

The StateInfo [▶ 598] type exposes the following members.

Methods

	Name	Description
	Equals(Object) [▶ 602]	Determines whether the specified Object is equal to this instance. (Overrides ValueType.Equals(Object).)

	Name	Description
	Equals(StateInfo) [▶ 603]	Determines whether the specified StateInfo [▶ 598] is equal to this instance.
	GetHashCode [▶ 603]	Returns a hash code for this instance. (Overrides ValueType.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns the fully qualified type name of this instance. (Inherited from ValueType.)

Reference

StateInfo Structure [▶ 598]

TwinCAT.Ads Namespace [▶ 108]

5.2.58.3.1 StateInfo.Equals Method

Overload List

	Name	Description
	Equals(Object) [▶ 602]	Determines whether the specified Object is equal to this instance. (Overrides ValueType.Equals(Object).)
	Equals(StateInfo) [▶ 603]	Determines whether the specified StateInfo [▶ 598] is equal to this instance.

Reference

StateInfo Structure [▶ 598]

TwinCAT.Ads Namespace [▶ 108]

StateInfo.Equals Method (Object)

Determines whether the specified Object is equal to this instance.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool Equals(
    Object ob
)
```

VB

```
Public Overrides Function Equals (
    ob As Object
) As Boolean
```

Parameters

ob	Type: System.Object The Object to compare with this instance.

Return Value

Type: Boolean
true if the specified Object is equal to this instance; otherwise, false.

Reference

StateInfo Structure [▶ 598]

Equals Overload [▶ 602]

TwinCAT.Ads Namespace [▶ 108]

StateInfo.Equals Method (StateInfo)

Determines whether the specified StateInfo [▶ 598] is equal to this instance.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Equals(
    StateInfo info
)
```

VB

```
Public Function Equals (
    info As StateInfo
) As Boolean
```

Parameters

info	Type: TwinCAT.Ads.StateInfo [▶ 598] The StateInfo [▶ 598] to compare with this instance.

Return Value

Type: Boolean
true if the specified StateInfo [▶ 598] is equal to this instance; otherwise, false.

Reference

StateInfo Structure [▶ 598]

Equals Overload [▶ 602]

TwinCAT.Ads Namespace [▶ 108]

5.2.58.3.2 StateInfo.GetHashCode Method

Returns a hash code for this instance.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override int GetHashCode()
```

VB

```
Public Overrides Function GetHashCode As Integer
```

Return Value

Type: Int32
A hash code for this instance, suitable for use in hashing algorithms and data structures like a hash table.

Reference

StateInfo Structure [▶ 598]

TwinCAT.Ads Namespace [▶ 108]

5.2.58.4 StateInfo Operators

The StateInfo [▶ 598] type exposes the following members.

Operators

	Name	Description
	Equality [▶ 604]	Implements the ==.
	Inequality [▶ 605]	Implements the !=.

Reference

StateInfo Structure [▶ 598]

TwinCAT.Ads Namespace [▶ 108]

5.2.58.4.1 StateInfo.Equality Operator

Implements the ==.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator ==(
    StateInfo a,
    StateInfo b
)
```

VB

```
Public Shared Operator = (
    a As StateInfo,
    b As StateInfo
) As Boolean
```

Parameters

a	Type: TwinCAT.Ads.StateInfo [▶ 598] a.
b	Type: TwinCAT.Ads.StateInfo [▶ 598] The b.

Return Value

Type: Boolean
The result of the operator.

Reference

StateInfo Structure [▶ 598]

TwinCAT.Ads Namespace [▶ 108]

5.2.58.4.2 StateInfo.Inequality Operator

Implements the !=.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator !=(
    StateInfo a,
    StateInfo b
)
```

VB

```
Public Shared Operator <> (
    a As StateInfo,
    b As StateInfo
) As Boolean
```

Parameters

a	Type: TwinCAT.Ads.StateInfo [▶ 598] a.
b	Type: TwinCAT.Ads.StateInfo [▶ 598] The b.

Return Value

Type: Boolean
The result of the operator.

Reference

StateInfo Structure [▶ 598]

TwinCAT.Ads Namespace [▶ 108]

5.2.59 SymbolException Class

Symbol bound exceptions

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.Ads.SymbolException
          TwinCAT.CannotAccessVirtualSymbolException [▶ 31]
          TwinCAT.InsufficientAccessRights [▶ 56]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class SymbolException : AdsException
```

VB

```
<SerializableAttribute>
Public Class SymbolException
    Inherits AdsException
```

The SymbolException type exposes the following members.

Constructors

	Name	Description
	SymbolException(ISymbol) [▶ 609]	Initializes a new instance of the SymbolException class.
	SymbolException(String, ISymbol) [▶ 609]	Initializes a new instance of the SymbolException class.
	SymbolException(ISymbol, Exception) [▶ 610]	Initializes a new instance of the SymbolException class.
	SymbolException(ISymbol, Int32) [▶ 611]	Initializes a new instance of the SymbolException class.

	Name	Description
	SymbolException(ISymbol, AdsErrorCode) [▶ 611]	Initializes a new instance of the SymbolException class.
	SymbolException(String, ISymbol, Exception) [▶ 612]	Initializes a new instance of the SymbolException class.
	SymbolException(ISymbol, AdsErrorCode, Exception) [▶ 613]	Initializes a new instance of the SymbolException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)

	Name	Description
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Fields

	Name	Description
	Symbol [▶ 615]	Symbol that is bound to the SymbolException

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.59.1 SymbolException Constructor

Overload List

	Name	Description
	SymbolException(ISymbol) [▶ 609]	Initializes a new instance of the SymbolException [▶ 606] class.
	SymbolException(String, ISymbol) [▶ 609]	Initializes a new instance of the SymbolException [▶ 606] class.
	SymbolException(ISymbol, Exception) [▶ 610]	Initializes a new instance of the SymbolException [▶ 606] class.
	SymbolException(ISymbol, Int32) [▶ 611]	Initializes a new instance of the SymbolException [▶ 606] class.
	SymbolException(ISymbol, AdsErrorCode) [▶ 611]	Initializes a new instance of the SymbolException [▶ 606] class.
	SymbolException(String, ISymbol, Exception) [▶ 612]	Initializes a new instance of the SymbolException [▶ 606] class.

	Name	Description
	SymbolException(ISymbol, AdsErrorCode, Exception) [▶ 613]	Initializes a new instance of the SymbolException [▶ 606] class.

Reference

SymbolException Class [▶ 606]

TwinCAT.Ads Namespace [▶ 108]

5.2.59.1.1 SymbolException Constructor (ISymbol)

Initializes a new instance of the SymbolException [▶ 606] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolException(
    ISymbol symbol
)
```

VB

```
Public Sub New (
    symbol As ISymbol
)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The symbol.

Reference

SymbolException Class [▶ 606]

SymbolException Overload [▶ 608]

TwinCAT.Ads Namespace [▶ 108]

5.2.59.1.2 SymbolException Constructor (String, ISymbol)

Initializes a new instance of the SymbolException [▶ 606] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolException(
    string message,
    ISymbol symbol
)
```

VB

```
Public Sub New (
    message As String,
    symbol As ISymbol
)
```

Parameters

message	Type: System.String The message.
symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The symbol.

Reference

SymbolException Class [▶ 606]

SymbolException Overload [▶ 608]

TwinCAT.Ads Namespace [▶ 108]

5.2.59.1.3 SymbolException Constructor (ISymbol, Exception)

Initializes a new instance of the SymbolException [▶ 606] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolException(
    ISymbol symbol,
    Exception innerException
)
```

VB

```
Public Sub New (
    symbol As ISymbol,
    innerException As Exception
)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The symbol.
innerException	Type: System.Exception The inner exception.

Reference

SymbolException Class [▶ 606]

SymbolException Overload [▶ 608]

TwinCAT.Ads Namespace [▶ 108]

5.2.59.1.4 SymbolException Constructor (ISymbol, Int32)

Initializes a new instance of the SymbolException [▶ 606] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolException(
    ISymbol symbol,
    int errorCode
)
```

VB

```
Public Sub New (
    symbol As ISymbol,
    errorCode As Integer
)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The symbol.
errorCode	Type: System.Int32 The error code.

Reference

SymbolException Class [▶ 606]

SymbolException Overload [▶ 608]

TwinCAT.Ads Namespace [▶ 108]

5.2.59.1.5 SymbolException Constructor (ISymbol, AdsErrorCode)

Initializes a new instance of the SymbolException [▶ 606] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolException(
    ISymbol symbol,
    AdsErrorCode errorCode
)
```

VB

```
Public Sub New (
    symbol As ISymbol,
    errorCode As AdsErrorCode
)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The symbol.
errorCode	Type: TwinCAT.Ads.AdsErrorCode [▶ 305] The error code.

Reference

SymbolException Class [▶ 606]

SymbolException Overload [▶ 608]

TwinCAT.Ads Namespace [▶ 108]

5.2.59.1.6 SymbolException Constructor (String, ISymbol, Exception)

Initializes a new instance of the SymbolException [▶ 606] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolException(
    string message,
    ISymbol symbol,
    Exception innerException
)
```

VB

```
Public Sub New (
    message As String,
    symbol As ISymbol,
    innerException As Exception
)
```

Parameters

message	Type: System.String The message.
symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The symbol.
innerException	Type: System.Exception The inner exception.

Reference

SymbolException Class [▶ 606]

SymbolException Overload [▶ 608]

TwinCAT.Ads Namespace [▶ 108]

5.2.59.1.7 SymbolException Constructor (ISymbol, AdsErrorCode, Exception)

Initializes a new instance of the SymbolException [▶ 606] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolException(
    ISymbol symbol,
    AdsErrorCode errorCode,
    Exception innerException
)
```

VB

```
Public Sub New (
    symbol As ISymbol,
    errorCode As AdsErrorCode,
    innerException As Exception
)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The symbol.
errorCode	Type: TwinCAT.Ads.AdsErrorCode [▶ 305] The error code.
innerException	Type: System.Exception The inner exception.

Reference

SymbolException Class [▶ 606]

SymbolException Overload [▶ 608]

TwinCAT.Ads Namespace [▶ 108]

5.2.59.2 SymbolException Properties

The SymbolException [▶ 606] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)

	Name	Description
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

SymbolException Class [▶ 606]

TwinCAT.Ads Namespace [▶ 108]

5.2.59.3 SymbolException Methods

The SymbolException [▶ 606] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

SymbolException Class [▶ 606]

TwinCAT.Ads Namespace [▶ 108]

5.2.59.4 SymbolException Events

The SymbolException [▶ 606] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

SymbolException Class [▶ 606]

TwinCAT.Ads Namespace [▶ 108]

5.2.59.5 SymbolException Fields

The SymbolException [▶ 606] type exposes the following members.

Fields

	Name	Description
	Symbol [▶ 615]	Symbol that is bound to the SymbolException [▶ 606]

Reference

SymbolException Class [▶ 606]

TwinCAT.Ads Namespace [▶ 108]

5.2.59.5.1 SymbolException.Symbol Field

Symbol that is bound to the SymbolException [▶ 606]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[NonSerializedAttribute]
public readonly ISymbol Symbol
```

VB

```
<NonSerializedAttribute>
Public ReadOnly Symbol As ISymbol
```

Field Value

Type: ISymbol [▶ 1634]

Reference

SymbolException Class [▶ 606]

TwinCAT.Ads Namespace [▶ 108]

5.2.60 SymbolLoaderSettings Class

Settings object for the IAdsSymbolLoader [▶ 979] initialization.

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.SymbolLoaderSettings
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SymbolLoaderSettings : ISymbolLoaderSettings
```

VB

```
Public Class SymbolLoaderSettings
    Implements ISymbolLoaderSettings
```

The SymbolLoaderSettings type exposes the following members.

Constructors

	Name	Description
	SymbolLoaderSettings(SymbolsLoadMode) [▶ 618]	Initializes a new instance of the SymbolLoaderSettings class with IndexGroupOffsetPreferred [▶ 1120].
	SymbolLoaderSettings(SymbolsLoadMode, ValueAccessMode) [▶ 618]	Initializes a new instance of the SymbolLoaderSettings class.
	SymbolLoaderSettings(SymbolsLoadMode, ValueCreationMode, ValueAccessMode) [▶ 619]	Initializes a new instance of the SymbolLoaderSettings class.

Properties

	Name	Description
	AutomaticReconnection [▶ 620]	Gets or sets a value indicating whether Disconnect connections can be reconnected.
S	Default [▶ 621]	Gets the default settings object (SymbolsLoadMode.VirtualTree + ValueAccessMode.IndexGroupOffsetPreferred)
S	DefaultDynamic [▶ 622]	Gets the default settings object (SymbolsLoadMode.DynamicTree + ValueCreationMode.TranslateToPrimitives + ValueAccessMode.IndexGroupOffsetPreferred)
	NonCachedArrayElements [▶ 622]	Gets or sets the setting to create ArrayElements "On-The-Fly" (Default True)

	Name	Description
	SymbolsLoadMode [▶ 623]	Gets or sets the symbols load mode.
	ValueAccessMode [▶ 623]	Gets or sets the value access mode.
	ValueCreation [▶ 624]	Gets or sets the value creation mode.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Remarks

This settings object is used for the initialization of the Symbol loader [▶ 979] object.

Reference

TwinCAT.Ads Namespace [▶ 108]

TwinCAT.ISymbolLoaderSettings [▶ 66]

TwinCAT.Ads.TypeSystem.SymbolLoaderFactory [▶ 1100]

TwinCAT.Ads.TypeSystem.IAdsSymbolLoader [▶ 979]

TwinCAT.SymbolsLoadMode [▶ 108]

TwinCAT.Ads.ValueAccess.ValueAccessMode [▶ 1120]

5.2.60.1 SymbolLoaderSettings Constructor

Overload List

	Name	Description
	SymbolLoaderSettings(SymbolsLoadMode) [▶ 618]	Initializes a new instance of the SymbolLoaderSettings [▶ 615] class with IndexGroupOffsetPreferred [▶ 1120].
	SymbolLoaderSettings(SymbolsLoadMo	Initializes a new instance of the SymbolLoaderSettings [▶ 615] class.

	Name	Description
	de, ValueAccessMode) [▶ 618]	
	SymbolLoaderSettings(SymbolsLoadMode, ValueCreationMode, ValueAccessMode) [▶ 619]	Initializes a new instance of the SymbolLoaderSettings [▶ 615] class.

Reference

SymbolLoaderSettings Class [▶ 615]

TwinCAT.Ads Namespace [▶ 108]

5.2.60.1.1 SymbolLoaderSettings Constructor (SymbolsLoadMode)

Initializes a new instance of the SymbolLoaderSettings [▶ 615] class with IndexGroupOffsetPreferred [▶ 1120].

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolLoaderSettings(
    SymbolsLoadMode loadMode
)
```

VB

```
Public Sub New (
    loadMode As SymbolsLoadMode
)
```

Parameters

loadMode	Type: TwinCAT.SymbolsLoadMode [▶ 108] The load mode.

Reference

SymbolLoaderSettings Class [▶ 615]

SymbolLoaderSettings Overload [▶ 617]

TwinCAT.Ads Namespace [▶ 108]

5.2.60.1.2 SymbolLoaderSettings Constructor (SymbolsLoadMode, ValueAccessMode)

Initializes a new instance of the SymbolLoaderSettings [▶ 615] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolLoaderSettings(
    SymbolsLoadMode loadMode,
    ValueAccessMode valueAccess
)
```

VB

```
Public Sub New (
    loadMode As SymbolsLoadMode,
    valueAccess As ValueAccessMode
)
```

Parameters

loadMode	Type: TwinCAT.SymbolsLoadMode [▶ 108] The load mode.
valueAccess	Type: TwinCAT.Ads.ValueAccess.ValueAccessMode [▶ 1120] The value access.

Reference

SymbolLoaderSettings Class [▶ 615]

SymbolLoaderSettings Overload [▶ 617]

TwinCAT.Ads Namespace [▶ 108]

5.2.60.1.3 SymbolLoaderSettings Constructor (SymbolsLoadMode, ValueCreationMode, ValueAccessMode)

Initializes a new instance of the SymbolLoaderSettings [▶ 615] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolLoaderSettings(
    SymbolsLoadMode loadMode,
    ValueCreationMode valueCreation,
    ValueAccessMode valueAccess
)
```

VB

```
Public Sub New (
    loadMode As SymbolsLoadMode,
    valueCreation As ValueCreationMode,
    valueAccess As ValueAccessMode
)
```

Parameters

loadMode	Type: TwinCAT.SymbolsLoadMode [▶ 108] The load mode.
valueCreation	Type: TwinCAT.ValueAccess.ValueCreationMode [▶ 1979] The dynamic value creation.

valueAccess	Type: TwinCAT.Ads.ValueAccess.ValueAccessMode [▶ 1120] The value access.

Reference

SymbolLoaderSettings Class [▶ 615]

SymbolLoaderSettings Overload [▶ 617]

TwinCAT.Ads Namespace [▶ 108]

5.2.60.2 SymbolLoaderSettings Properties

The SymbolLoaderSettings [▶ 615] type exposes the following members.

Properties

	Name	Description
	AutomaticReconnection [▶ 620]	Gets or sets a value indicating whether Disconnect connections can be reconnected.
S	Default [▶ 621]	Gets the default settings object (SymbolsLoadMode.VirtualTree + ValueAccessMode.IndexGroupOffsetPreferred)
S	DefaultDynamic [▶ 622]	Gets the default settings object (SymbolsLoadMode.DynamicTree + ValueCreationMode.TranslateToPrimitives + ValueAccessMode.IndexGroupOffsetPreferred)
	NonCachedArrayElements [▶ 622]	Gets or sets the setting to create ArrayElements "On-The-Fly" (Default True)
	SymbolsLoadMode [▶ 623]	Gets or sets the symbols load mode.
	ValueAccessMode [▶ 623]	Gets or sets the value access mode.
	ValueCreation [▶ 624]	Gets or sets the value creation mode.

Reference

SymbolLoaderSettings Class [▶ 615]

TwinCAT.Ads Namespace [▶ 108]

5.2.60.2.1 SymbolLoaderSettings.AutomaticReconnection Property

Gets or sets a value indicating whether Disconnect connections can be reconnected.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool AutomaticReconnection { get; set; }
```

VB

```
Public Property AutomaticReconnection As Boolean
    Get
    Set
```

Property Value

Type: Boolean
true if Disconnect connections can be reconnecte; otherwise, false.

Reference

SymbolLoaderSettings Class [▶ 615]

TwinCAT.Ads Namespace [▶ 108]

5.2.60.2.2 SymbolLoaderSettings.Default Property

Gets the default settings object (SymbolsLoadMode.VirtualTree + ValueAccessMode.IndexGroupOffsetPreferred)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static SymbolLoaderSettings Default { get; }
```

VB

```
Public Shared ReadOnly Property Default As SymbolLoaderSettings
    Get
```

Property Value

Type: SymbolLoaderSettings [▶ 615]
The default settings object.

Remarks

The following defaults are set here:

Setting	Description
Symbols load mode (SymbolsLoadMode [▶ 623])	Create virtual tree (VirtualTree [▶ 108].
Value access mode (ValueAccessMode [▶ 623])	Prefer IndexGroup/IndexOffset communication if available (IndexGroupOffsetPreferred [▶ 1120]).
Value creation mode ValueCreation [▶ 624]	Create primitives if possible (Default [▶ 1979].

Reference

SymbolLoaderSettings Class [▶ 615]

TwinCAT.Ads Namespace [▶ 108]

5.2.60.2.3 SymbolLoaderSettings.DefaultDynamic Property

Gets the default settings object (SymbolsLoadMode.DynamicTree + ValueCreationMode.TranslateToPrimitives + ValueAccessMode.IndexGroupOffsetPreferred)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static SymbolLoaderSettings DefaultDynamic { get; }
```

VB

```
Public Shared ReadOnly Property DefaultDynamic As SymbolLoaderSettings
    Get
```

Property Value

Type: SymbolLoaderSettings [▶ 615]
The dynamic default settings object.

Remarks

The following defaults are set here:

Setting	Description
Symbols load mode (SymbolsLoadMode [▶ 623])	Create dynamic tree (DynamicTree [▶ 108].
Value access mode (ValueAccessMode [▶ 623])	Prefer IndexGroup/IndexOffset communication if available (IndexGroupOffsetPreferred [▶ 1120]).
Value creation mode ValueCreation [▶ 624]	Create primitives if possible (Default [▶ 1979].

Reference

SymbolLoaderSettings Class [▶ 615]

TwinCAT.Ads Namespace [▶ 108]

5.2.60.2.4 SymbolLoaderSettings.NonCachedArrayElements Property

Gets or sets the setting to create ArrayElements "On-The-Fly" (Default True)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool NonCachedArrayElements { get; set; }
```

VB

```
Public Property NonCachedArrayElements As Boolean
    Get
    Set
```

Property Value

Type: Boolean
The value access mode.

Reference

SymbolLoaderSettings Class [▶ 615]

TwinCAT.Ads Namespace [▶ 108]

5.2.60.2.5 SymbolLoaderSettings.SymbolsLoadMode Property

Gets or sets the symbols load mode.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolsLoadMode SymbolsLoadMode { get; set; }
```

VB

```
Public Property SymbolsLoadMode As SymbolsLoadMode
    Get
    Set
```

Property Value

Type: SymbolsLoadMode [▶ 108]
The symbols load mode.

Reference

SymbolLoaderSettings Class [▶ 615]

TwinCAT.Ads Namespace [▶ 108]

5.2.60.2.6 SymbolLoaderSettings.ValueAccessMode Property

Gets or sets the value access mode.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ValueAccessMode ValueAccessMode { get; set; }
```

VB

```
Public Property ValueAccessMode As ValueAccessMode
    Get
    Set
```

Property Value

Type: ValueAccessMode [▶ 1120]
The value access mode.

Reference

SymbolLoaderSettings Class [▶ 615]

TwinCAT.Ads Namespace [▶ 108]

5.2.60.2.7 SymbolLoaderSettings.ValueCreation Property

Gets or sets the value creation mode.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ValueCreationMode ValueCreation { get; set; }
```

VB

```
Public Property ValueCreation As ValueCreationMode
    Get
    Set
```

Property Value

Type: ValueCreationMode [▶ 1979]
The dynamic value mode.

Reference

SymbolLoaderSettings Class [▶ 615]

TwinCAT.Ads Namespace [▶ 108]

5.2.60.3 SymbolLoaderSettings Methods

The SymbolLoaderSettings [▶ 615] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)

	Name	Description
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

SymbolLoaderSettings Class [▶ 615]

TwinCAT.Ads Namespace [▶ 108]

5.2.61 TcAdsClient Class

ADS Client / ADS Communication object.

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TcAdsClient
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class TcAdsClient : ITcAdsRpcInvoke,
    IDisposable, IAdsConnection, IConnection, IConnectionStateProvider, IAdsNotifications,
    IAdsAnyAccess, IAdsHandleAccess
```

VB

```
Public Class TcAdsClient
    Implements ITcAdsRpcInvoke, IDisposable, IAdsConnection, IConnection,
    IConnectionStateProvider, IAdsNotifications, IAdsAnyAccess, IAdsHandleAccess
```

The TcAdsClient type exposes the following members.

Constructors

	Name	Description
	TcAdsClient. [▶ 639]	Initializes a new instance of the class TcAdsClient.
	TcAdsClient(AdsClientSettings) [▶ 640]	Initializes a new instance of the TcAdsClient class.

Properties

	Name	Description
	Address [▶ 641]	Gets the AmsAddress [▶ 410] of the ADS server (Target side)
	ClientAddress [▶ 641]	Get the AmsAddress [▶ 410] of the ADS client (Source side)
	Disposed [▶ 642]	Determines, whether the TcAdsClient is disposed

	Name	Description
	Id [▶ 643]	Gets the TcAdsClient Identifier.
	IsConnected [▶ 643]	Gets a value indicating whether the local ADS port was opened successfully. It does not indicate if the target port is available. Use the method ReadState to determine if the target port is available.
	IsLocal [▶ 644]	Gets a value indicating whether the ADS client is connected to a ADS Server on the local computer.
	Protocol [▶ 644]	Gets actual ADS Transport protocol
	RouterState [▶ 645]	Gets the current state of the local AMS Router.
	Session [▶ 645]	Gets the session that initiated this IConnection [▶ 46]
	Synchronize [▶ 646]	Gets or sets a value indicating whether the TcAdsClient object synchronizes the incoming notifications (obsolete)
	Timeout [▶ 647]	Sets the timeout for the ads communication. Unit is in ms.

Methods

	Name	Description
	AddDeviceNotification(String, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 659]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String, AdsStream, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 660]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 661]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 663]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String,	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 665]	
	AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 667]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 669]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 671]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type) [▶ 674]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, TimeSpan, TimeSpan, Object, Type) [▶ 676]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 678]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AddDeviceNotificationEx(String, AdsTransMode, TimeSpan, TimeSpan, Object, Type, .Int32.) [▶ 680]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type) [▶ 682]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, TimeSpan, TimeSpan, Object, Type) [▶ 683]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 685]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, TimeSpan, TimeSpan, Object, Type, .Int32.) [▶ 687]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	Close [▶ 689]	Closes this IConnection [▶ 46]
	Connect(Int32) [▶ 690]	Establishes a connection to a ADS device using the local netID.
	Connect(AmsAddress) [▶ 690]	Establishes a connection to a ADS device.
	Connect(.Byte., Int32) [▶ 691]	Establishes a connection to a ADS device.
	Connect(String, Int32) [▶ 691]	Establishes a connection to a ADS device.
	Connect(AmsNetId, Int32) [▶ 692]	Establishes a connection to a ADS device.
	Connect(AmsNetId, AmsPort) [▶ 693]	Establishes a connection to a ADS device.
	CreateSymbolInfoLoader [▶ 693]	**Obsolete.** Creates a new instance of the TcAdsSymbolInfoLoader class (Symbol Browser V1, obsolete).

	Name	Description
	CreateVariableHandle [▶ 696]	Generates a unique handle for an ADS variable.
	DeleteDeviceNotification [▶ 697]	Deletes an existing notification.
	DeleteVariableHandle [▶ 698]	Releases the handle of a ADS variable again.
	Disconnect [▶ 699]	Disconnects the TcAdsClient
	Dispose. [▶ 700]	Releases the resources used by TcAdsClient.
	Dispose(Boolean) [▶ 700]	Disposes the TcAdsClient.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize [▶ 701]	Finalizes an instance of the TcAdsClient class. (Overrides Object.Finalize..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	InvokeRpcMethod(String, Int32, .Object.) [▶ 702]	Invokes the specified RPC Method
	InvokeRpcMethod(String, String, .Object.) [▶ 703]	Invokes the RPC method.
	InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 705]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 706]	Invokes the RPC method.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnBeforeDisconnect [▶ 707]	Called when before the TcAdsClient is disconnected.
	OnConnectionStateChanged [▶ 708]	Called when the ConnectionState of the TcAdsClient has changed.
	Read(Int32, AdsStream) [▶ 709]	Reads data synchronously from an ADS device and writes it to the given stream.

	Name	Description
	Read(UInt32, UInt32, AdsStream) [▶ 710]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(Int32, AdsStream, Int32, Int32) [▶ 711]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, .Byte., Int32, Int32) [▶ 712]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 713]	Reads data synchronously from an ADS device and writes it to the given stream.
	ReadAny(Int32, Type) [▶ 714]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(Int32, Type, .Int32.) [▶ 715]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type) [▶ 716]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 717]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAnyString(Int32, Int32, Encoding) [▶ 718]	Reads the string.
	ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 720]	Reads the string
	ReadDeviceInfo [▶ 721]	Reads the identification and version number of an ADS server.
	ReadState [▶ 722]	Reads the ADS status and the device status from an ADS server.
	ReadSymbol(ITcAdsSymbol) [▶ 722]	Reads the value of a symbol and returns it as an object. Strings and all primitive datatypes(UInt32, Int32, Bool etc.) are supported. Arrays and structures cannot be read.
	ReadSymbol(String, Type, Boolean) [▶ 723]	Reads the value of a symbol and returns the value as object. The parameter type must have the same layout as the ADS symbol.
	ReadSymbolInfo [▶ 724]	Call this method to obtain information about the individual symbols (variables) in ADS devices.
	ReadWrite(UInt32, UInt32, AdsStream, AdsStream) [▶ 725]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 726]	Writes data synchronously to an ADS device and then Reads data from this device.

	Name	Description
	ReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32) [▶ 727]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 729]	Writes data synchronously to an ADS device and then Reads data from this device.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryAddDeviceNotification [▶ 730]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	TryAddDeviceNotificationEx [▶ 731]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	TryDeleteDeviceNotification [▶ 732]	Deletes an existing notification.
	TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 734]	Invokes the specified RPC Method
	TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 735]	Tries to invoke the RPC method.
	TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 736]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 737]	Tries to invoke a RPC method.
	TryRead(UInt32, UInt32, AdsStream, Int32.) [▶ 739]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(Int32, AdsStream, Int32, Int32, Int32.) [▶ 739]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, .Byte., Int32, Int32, Int32.) [▶ 740]	Reads data synchronously from an ADS device and writes it to the given stream.

	Name	Description
	TryRead(UInt32, UInt32, AdsStream, Int32, Int32, Int32.) [▶ 741]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryReadState [▶ 742]	Reads the ADS status and the device status from an ADS server. Unlike the ReadState method this method does not call an exception on failure. Instead an AdsErrorCode is returned. If the return value is equal to AdsErrorCode.NoError the call was successfull.
	TryReadWrite(Int32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32.) [▶ 744]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 745]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32.) [▶ 746]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 747]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryWrite(Int32, AdsStream, Int32, Int32) [▶ 749]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, .Byte., Int32, Int32) [▶ 750]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 751]	Writes data synchronously to an ADS device.
	TryWriteControl(StateInfo) [▶ 752]	Changes the ADS status and the device status of an ADS server.
	TryWriteControl(StateInfo, AdsStream, Int32, Int32) [▶ 753]	Changes the ADS status and the device status of an ADS server.
	Write(Int32, Int32) [▶ 754]	Trigger Client Method/Command.
	Write(Int32, AdsStream) [▶ 755]	Writes data synchronously to an ADS device.

	Name	Description
	Write(UInt32, UInt32) [▶ 756]	Trigger Client Method/Command.
	Write(Int32, Int32, AdsStream) [▶ 757]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream) [▶ 758]	Writes data synchronously to an ADS device.
	Write(Int32, AdsStream, Int32, Int32) [▶ 758]	Writes data synchronously to an ADS device.
	Write(Int32, Int32, .Byte., Int32, Int32) [▶ 759]	Writes data synchronously to an ADS device.
	Write(Int32, Int32, AdsStream, Int32, Int32) [▶ 760]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, .Byte., Int32, Int32) [▶ 761]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 762]	Writes data synchronously to an ADS device.
	WriteAny(Int32, Object) [▶ 763]	Writes an object synchronously to an ADS device.
	WriteAny(Int32, Object, .Int32.) [▶ 764]	Writes an object synchronously to an ADS device.
	WriteAny(UInt32, UInt32, Object) [▶ 765]	Writes an object synchronously to an ADS device.
	WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 766]	Writes an object synchronously to an ADS device.
	WriteAnyString(Int32, String, Int32, Encoding) [▶ 767]	Writes the string (Potentially unsafe!)
	WriteAnyString(UInt32, UInt32, String, Int32, Encoding) [▶ 769]	Writes the string (Potentially unsafe!)
	WriteControl(StateInfo) [▶ 770]	Changes the ADS status and the device status of an ADS server.
	WriteControl(StateInfo, AdsStream, Int32, Int32) [▶ 771]	Changes the ADS status and the device status of an ADS server.

	Name	Description
	WriteSymbol(ITcAdsSymbol, Object) [▶ 772]	Writes a value to the symbol. Strings and all primitive data types(UInt32, Int32, Bool etc.) are supported. Array and structures are not supported. If a string is passed as parameter, the method attempts to parse the string according to the ADS data type of the symbol.
	WriteSymbol(String, Object, Boolean) [▶ 773]	Writes the passed object value to the specified ADS symbol.The parameter type must have the same layout as the ADS symbol.

Events

	Name	Description
	AdsNotification [▶ 774]	Occurs when the ADS device sends a ADS Notification to the client.
	AdsNotificationError [▶ 776]	Occurs when a exception has occurred during notification management.
	AdsNotificationEx [▶ 776]	Occurs when the ADS devices sends an (extended) notification to the client.
	AdsStateChanged [▶ 778]	Occurs when the ADS state changes.
	AdsSymbolVersionChanged [▶ 778]	Occurs when the symbol version has been changed changes.
	AmsRouterNotification [▶ 779]	Occurs when the state of the local Router has changed.
	ConnectionStateChanged [▶ 779]	Occurs when the connection state has been changed.

Extension Methods

	Name	Description
	PollAdsState(IObservable.Unit.) [▶ 825]	Overloaded. Gets an observable sequence of AdsState [▶ 365]s via Polling. (Defined by AdsClientExtensions [▶ 821].)
	PollAdsState(TimeSpan) [▶ 826]	Overloaded. Gets an observable sequence of AdsState [▶ 365]s via Polling. (Defined by AdsClientExtensions [▶ 821].)
	PollValues(String, Type, IObservable.Unit.) [▶ 849]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, TimeSpan) [▶ 850]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)

	Name	Description
	PollValues(String, Type, .Int32., TimeSpan) [▶ 853]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, IObservable.Unit., Func.Exception, Object.) [▶ 854]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, TimeSpan, Func.Exception, Object.) [▶ 855]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., IObservable.Unit., Func.Exception, Object.) [▶ 856]	Overloaded. Polls the symbol values on time points where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., TimeSpan, Func.Exception, Object.) [▶ 858]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, IObservable.Unit.) [▶ 842]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, TimeSpan) [▶ 843]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, IObservable.Unit., Func.Exception, T.) [▶ 847]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, TimeSpan, Func.Exception, T.) [▶ 848]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, .Int32., IObservable.Unit.) [▶ 844]	Overloaded. Polls the symbol values on time points where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, .Int32., TimeSpan) [▶ 845]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, .Int32., IObservable.Unit., Func.Exception, T.) [▶ 851]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)

	Name	Description
	PollValues.T. (String, .Int32., TimeSpan, Func.Exception, T.) [▶ 852]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	WhenAdsStateChanges [▶ 828]	Gets an observable sequence of AdsState [▶ 365]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbol) [▶ 829]	Overloaded. Gets an observable sequence of Notification [▶ 862]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbolCollection) [▶ 830]	Overloaded. Gets an observable sequence of Notification [▶ 862] objects. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbol, NotificationSettings) [▶ 832]	Overloaded. Gets an observable sequence of Notification [▶ 862]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbolCollection, NotificationSettings) [▶ 833]	Overloaded. Gets an observable sequence of Notification [▶ 862] objects. (Defined by AdsClientExtensions [▶ 821].)
	WhenValueChanged [▶ 886]	Observable sequence of Values driven by ADS Notifications on the specified symbol. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues.T. (String, IObservable.T.) [▶ 859]	Overloaded. Writes the sequence of values to the symbol specified by the instance path. (Defined by AnyTypeExtensions [▶ 835].)
	WriteValues.T. (String, IObservable.T., Action.Exception.) [▶ 860]	Overloaded. Writes the sequence of values to the symbol specified by the instance path. (Defined by AnyTypeExtensions [▶ 835].)

Remarks

IMPORTANT: The Default setting of the Synchronize [▶ 646] property has changed to 'false' from Version 4.2.XX on. This has the effect that - by default - the notifications events and are not synchronized into the UI thread anymore. To re enable the obsolete behavior set to 'true'.
AdsNotification [▶ 774] AdsNotificationEx [▶ 776] Synchronize [▶ 646] Use an instance of this object to create a point-to-point send/receive connection to an ADS Server/Device object. The class TcAdsClient is a wrapper for the TcAdsdll.dll and enables synchronous/asynchronous access to data of an ADS Device.

Examples

The following sample shows how to instantiate and use the TcAdsClient class.

TcAdsClient Demo

```
using System;
using System.Threading;

using TwinCAT.Ads;
using TwinCAT.TypeSystem;
using TwinCAT.TypeSystem.Generic;

namespace Sample
{
    class AdsClient
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        AmsAddress address = ArgParser.Parse(args);

        using (TcAdsClient client = new TcAdsClient())
        {
        // Asynchronized access necessary for Console applications
        client.Synchronize = false;

        // Connect to Address
        client.Connect(address.NetId,address.Port); // Connect to Port (851, first PLC by default)

        // Read the identification and version number of the device
        DeviceInfo deviceInfo = client.ReadDeviceInfo();
        Version version = deviceInfo.Version.ConvertToStandard();
        Console.WriteLine(string.Format("DeviceName: {0}", deviceInfo.Name));
        Console.WriteLine(string.Format("DeviceVersion: {0}", version.ToString(3)));

        // Read the state of the device
        StateInfo stateInfo = client.ReadState();
        AdsState state = stateInfo.AdsState;
        short deviceState = stateInfo.DeviceState;
        Console.WriteLine(string.Format("DeviceState: {0}", deviceState));
        Console.WriteLine(string.Format("AdsState   : {0}", state));

        // Write ADS Commands (write state) to target
        // Set PLC to Run

        if (state == AdsState.Stop)
        {
            StateInfo setState = new StateInfo(AdsState.Run, 0);
            client.WriteControl(setState);
        }

        //create variable handle for Plc Project Name (automatic generated symbol in PLC)
        int handleProjectName = client.CreateVariableHandle("TwinCAT_SystemInfoVarList._AppInfo.Proj
ectName");
        int handleNotification = 0; // Notification Handle for Task1 CycleCount changes

        try
        {
            // Read value from target and Marshal data into string
            using (AdsBinaryReader reader = new AdsBinaryReader(new AdsStream(256)))
            {
            int read = client.Read(handleProjectName, (AdsStream)reader.BaseStream);
            string projectName = reader.ReadPlcAnsiString(256);
            Console.WriteLine(string.Format("ProjectName  : {0}", projectName));
            }

            _notificationStream = new AdsStream(4); // Sizeof UDINT
            _notificationReader = new AdsBinaryReader(_notificationStream);

            client.AdsNotification += client_NotificationEvent; // Register for Notification event

            //Register Notification for Task1 CycleCount symbol (automatic generated symbol in PLC)
            handleNotification = client.AddDeviceNotification("TwinCAT_SystemInfoVarList._TaskInfo[1
].CycleCount", _notificationStream, AdsTransMode.OnChange, 500, 0, null);

            // Sleep 10 Seconds to receive events
            System.Threading.Thread.Sleep(10000);
        }
        finally
        {
```

```
            // Cleanup all handles
            // Dispose all Streams

            client.DeleteDeviceNotification(handleNotification);
            _notificationReader.Close();
            _notificationStream.Close();

            // Always delete all variable handles.
            client.DeleteVariableHandle(handleProjectName);
        }
        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
        }
    }
```

Argument Parser

```
public static class ArgParser
{
    /// <summary>
    /// Parses the arguments.
    /// </summary>
    /// <param name="args">The arguments.</param>
    /// <returns>AmsAddress.</returns>
    public static AmsAddress Parse(string[] args)
    {
    AmsNetId netId = AmsNetId.Local;
    int port = 851;

    if (args != null)
    {
        if (args.Length > 0 && args[0] != null)
        netId = AmsNetId.Parse(args[0]);

        if (args.Length > 1 && args[1] != null)
        port = int.Parse(args[1]);
    }
    return new AmsAddress(netId, port);
    }
}
```

The following sample shows how to call (Remote Procedures / Methods) within the PLC directly from the TcAdsClient class.

RPC Call Example

```
namespace Sample
{
    using System;
    using System.Diagnostics;
    using TwinCAT.Ads;
    using TwinCAT.TypeSystem;

    class RpcCallV1Program
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        //Parse the AmsAddress from command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        // Create the ADS Client
        using (TcAdsClient client = new TcAdsClient())
        {
        // Establish Connection
        client.Connect(address);

        // Call a Method that has the following signature (within MAIN Program)

        /*  {attribute 'TcRpcEnable'}
            METHOD PUBLIC M_Add : INT
            VAR_INPUT
            i1 : INT := 0;
            i2 : INT := 0;
```

```
            END_VAR
        */

        short result = (short)client.InvokeRpcMethod("MAIN", "M_Add", new object[] {(short)1, (short
)4});

        // Call a Method that has no parameter and returns VOID
        client.InvokeRpcMethod("MAIN", "M_Method1", new object[] {});
        }
    }
    }
}
```

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.61.1 TcAdsClient Constructor

Overload List

	Name	Description
	TcAdsClient. [▶ 639]	Initializes a new instance of the class TcAdsClient.
	TcAdsClient(AdsClientSettings) [▶ 640]	Initializes a new instance of the TcAdsClient [▶ 625] class.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.1.1 TcAdsClient Constructor

Initializes a new instance of the class TcAdsClient.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TcAdsClient()
```

VB

```
Public Sub New
```

Remarks

Default Interceptors (FailFastHandler [▶ 523]) will be used.

Reference

TcAdsClient Class [▶ 625]

TcAdsClient Overload [▶ 639]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.1.2 TcAdsClient Constructor (AdsClientSettings)

Initializes a new instance of the TcAdsClient [▶ 625] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TcAdsClient(
    AdsClientSettings settings
)
```

VB

```
Public Sub New (
    settings As AdsClientSettings
)
```

Parameters

settings	Type: TwinCAT.Ads.AdsClientSettings [▶ 133] The settings.

Reference

TcAdsClient Class [▶ 625]

TcAdsClient Overload [▶ 639]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.2 TcAdsClient Properties

The TcAdsClient [▶ 625] type exposes the following members.

Properties

	Name	Description
	Address [▶ 641]	Gets the AmsAddress [▶ 410] of the ADS server (Target side)
	ClientAddress [▶ 641]	Get the AmsAddress [▶ 410] of the ADS client (Source side)
	Disposed [▶ 642]	Determines, whether the TcAdsClient [▶ 625] is disposed
	Id [▶ 643]	Gets the TcAdsClient [▶ 625] Identifier.
	IsConnected [▶ 643]	Gets a value indicating whether the local ADS port was opened successfully. It does not indicate if the target port is available. Use the method ReadState to determine if the target port is available.
	IsLocal [▶ 644]	Gets a value indicating whether the ADS client is connected to a ADS Server on the local computer.
	Protocol [▶ 644]	Gets actual ADS Transport protocol
	RouterState [▶ 645]	Gets the current state of the local AMS Router.
	Session [▶ 645]	Gets the session that initiated this IConnection [▶ 46]

	Name	Description
	Synchronize [▶ 646]	Gets or sets a value indicating whether the TcAdsClient object synchronizes the incoming notifications (obsolete)
	Timeout [▶ 647]	Sets the timeout for the ads communication. Unit is in ms.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.2.1 TcAdsClient.Address Property

Gets the AmsAddress [▶ 410] of the ADS server (Target side)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress Address { get; }
```

VB

```
Public ReadOnly Property Address As AmsAddress
    Get
```

Property Value

Type: AmsAddress [▶ 410]

Implements

IAdsConnection.Address [▶ 477]

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.2.2 TcAdsClient.ClientAddress Property

Get the AmsAddress [▶ 410] of the ADS client (Source side)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress ClientAddress { get; }
```

VB

```
Public ReadOnly Property ClientAddress As AmsAddress
    Get
```

Property Value

Type: AmsAddress [▶ 410]
The client address if connection is up and running, otherwise **NULL**.

Implements

IAdsConnection.ClientAddress [▶ 477]

Remarks

The ClientAddress is only available if the connection is up and running. For the different (possible) connection methods:

Port	Description
1	Connection is using the TCP/IP protocol.
>=0x8000 (>=32768)	Ads communication via ADS Router (Default)

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.2.3 TcAdsClient.Disposed Property

Determines, whether the TcAdsClient [▶ 625] is disposed

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Disposed { get; }
```

VB

```
Public ReadOnly Property Disposed As Boolean
    Get
```

Property Value

Type: Boolean

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.2.4 TcAdsClient.Id Property

Gets the TcAdsClient [▶ 625] Identifier.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Id { get; }
```

VB

```
Public ReadOnly Property Id As Integer
    Get
```

Property Value

Type: Int32
The identifier.

Implements

IConnection.Id [▶ 48]

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.2.5 TcAdsClient.IsConnected Property

Gets a value indicating whether the local ADS port was opened successfully. It does not indicate if the target port is available. Use the method ReadState to determine if the target port is available.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsConnected { get; }
```

VB

```
Public ReadOnly Property IsConnected As Boolean
    Get
```

Property Value

Type: Boolean

Implements

IConnection.IsConnected [▶ 48]

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.2.6 TcAdsClient.IsLocal Property

Gets a value indicating whether the ADS client is connected to a ADS Server on the local computer.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsLocal { get; }
```

VB

```
Public ReadOnly Property IsLocal As Boolean
    Get
```

Property Value

Type: Boolean

Implements

IAdsConnection.IsLocal [▶ 477]

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.2.7 TcAdsClient.Protocol Property

Gets actual ADS Transport protocol

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TransportProtocol Protocol { get; }
```

VB

```
Public ReadOnly Property Protocol As TransportProtocol
    Get
```

Property Value

Type: TransportProtocol [▶ 820]
The protocol.

Remarks

Transport protocol	Description
TcpIp [▶ 820]	Connection is using the TCP/IP protocol (Client port 1)
Router [▶ 820]	Ads communication via local ADS Router (Default, Client Port >=0x8000 (32768))

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.2.8 TcAdsClient.RouterState Property

Gets the current state of the local AMS Router.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsRouterState RouterState { get; }
```

VB

```
Public ReadOnly Property RouterState As AmsRouterState
    Get
```

Property Value

Type: AmsRouterState [▶ 455]

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.2.9 TcAdsClient.Session Property

Gets the session that initiated this IConnection [▶ 46]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ISession Session { get; }
```

VB

```
Public ReadOnly Property Session As ISession
    Get
```

Property Value

Type: ISession [▶ 60]
The session or NULL

Implements

IConnection.Session [▶ 49]

Remarks

The Session can be null on standalone connections.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.2.10 TcAdsClient.Synchronize Property

Gets or sets a value indicating whether the TcAdsClient object synchronizes the incoming notifications (obsolete)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Synchronize { get; set; }
```

VB

```
Public Property Synchronize As Boolean
    Get
    Set
```

Property Value

Type: Boolean

Remarks

IMPORTANT: Be aware that this setting could be removed in later version of this API. The default setting changed from 'true' to 'false' in newer versions, because synchronization should be done in the Application and not has helper functionality within this class / API. Please think about of using .NET SynchronizationContexts or Control.Invoke(...) calls for Application side synchronization.
If Synchronize is set to true, the notifications are synchronized onto the Main thread. This is helpful for Windows Forms projects because this automatically synchronizes the notifications into the main UI thread and prevents any synchronization issues like deadlocks. In Console Applications it is necessary to set this flag to false if ADS notifications are used, because the Console Host doesn't trigger any message pumps, effectively blocking any Windows Messages and ADS Notifications.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.2.11 TcAdsClient.Timeout Property

Sets the timeout for the ads communication. Unit is in ms.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Timeout { get; set; }
```

VB

```
Public Property Timeout As Integer
    Get
    Set
```

Property Value

Type: Int32

Implements

IConnection.Timeout [▶ 49]

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3 TcAdsClient Methods

The TcAdsClient [▶ 625] type exposes the following members.

Methods

	Name	Description
	AddDeviceNotification(String, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 659]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String, AdsStream, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 660]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 661]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 663]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 665]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 667]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 669]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 671]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String,	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AdsTransMode, Int32, Int32, Object, Type) [▶ 674]	
	AddDeviceNotificationEx(String, AdsTransMode, TimeSpan, TimeSpan, Object, Type) [▶ 676]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 678]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, TimeSpan, TimeSpan, Object, Type, .Int32.) [▶ 680]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type) [▶ 682]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, TimeSpan, TimeSpan, Object, Type) [▶ 683]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 685]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, TimeSpan, TimeSpan, Object, Type, .Int32.) [▶ 687]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	Close [▶ 689]	Closes this IConnection [▶ 46]

	Name	Description
	Connect(Int32) [▶ 690]	Establishes a connection to a ADS device using the local netID.
	Connect(AmsAddress) [▶ 690]	Establishes a connection to a ADS device.
	Connect(.Byte., Int32) [▶ 691]	Establishes a connection to a ADS device.
	Connect(String, Int32) [▶ 691]	Establishes a connection to a ADS device.
	Connect(AmsNetId, Int32) [▶ 692]	Establishes a connection to a ADS device.
	Connect(AmsNetId, AmsPort) [▶ 693]	Establishes a connection to a ADS device.
	CreateSymbolInfoLoader [▶ 693]	**Obsolete.** Creates a new instance of the TcAdsSymbolInfoLoader class (Symbol Browser V1, obsolete).
	CreateVariableHandle [▶ 696]	Generates a unique handle for an ADS variable.
	DeleteDeviceNotification [▶ 697]	Deletes an existing notification.
	DeleteVariableHandle [▶ 698]	Releases the handle of a ADS variable again.
	Disconnect [▶ 699]	Disconnects the TcAdsClient [▶ 625]
	Dispose. [▶ 700]	Releases the resources used by TcAdsClient.
	Dispose(Boolean) [▶ 700]	Disposes the TcAdsClient [▶ 625].
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize [▶ 701]	Finalizes an instance of the TcAdsClient [▶ 625] class. (Overrides Object.Finalize..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	InvokeRpcMethod(String, Int32, .Object.) [▶ 702]	Invokes the specified RPC Method
	InvokeRpcMethod(String, String, .Object.) [▶ 703]	Invokes the RPC method.

	Name	Description
	InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 705]	Invokes the specified RPC Method
	InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 706]	Invokes the RPC method.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnBeforeDisconnect [▶ 707]	Called when before the TcAdsClient [▶ 625] is disconnected.
	OnConnectionStateChanged [▶ 708]	Called when the ConnectionState of the TcAdsClient [▶ 625] has changed.
	Read(Int32, AdsStream) [▶ 709]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, AdsStream) [▶ 710]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(Int32, AdsStream, Int32, Int32) [▶ 711]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, .Byte., Int32, Int32) [▶ 712]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 713]	Reads data synchronously from an ADS device and writes it to the given stream.
	ReadAny(Int32, Type) [▶ 714]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(Int32, Type, .Int32.) [▶ 715]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type) [▶ 716]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 717]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAnyString(Int32, Int32, Encoding) [▶ 718]	Reads the string.
	ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 720]	Reads the string

	Name	Description
	ReadDeviceInfo [▶ 721]	Reads the identification and version number of an ADS server.
	ReadState [▶ 722]	Reads the ADS status and the device status from an ADS server.
	ReadSymbol(ITcAdsSymbol) [▶ 722]	Reads the value of a symbol and returns it as an object. Strings and all primitive datatypes(UInt32, Int32, Bool etc.) are supported. Arrays and structures cannot be read.
	ReadSymbol(String, Type, Boolean) [▶ 723]	Reads the value of a symbol and returns the value as object. The parameter type must have the same layout as the ADS symbol.
	ReadSymbolInfo [▶ 724]	Call this method to obtain information about the individual symbols (variables) in ADS devices.
	ReadWrite(UInt32, UInt32, AdsStream, AdsStream) [▶ 725]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 726]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32) [▶ 727]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 729]	Writes data synchronously to an ADS device and then Reads data from this device.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryAddDeviceNotification [▶ 730]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	TryAddDeviceNotificationEx [▶ 731]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	TryDeleteDeviceNotification [▶ 732]	Deletes an existing notification.
	TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 734]	Invokes the specified RPC Method
	TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 735]	Tries to invoke the RPC method.

	Name	Description
	TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 736]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 737]	Tries to invoke a RPC method.
	TryRead(UInt32, UInt32, AdsStream, Int32.) [▶ 739]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(Int32, AdsStream, Int32, Int32, Int32.) [▶ 739]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, .Byte., Int32, Int32, Int32.) [▶ 740]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, AdsStream, Int32, Int32, Int32.) [▶ 741]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryReadState [▶ 742]	Reads the ADS status and the device status from an ADS server. Unlike the ReadState method this method does not call an exception on failure. Instead an AdsErrorCode is returned. If the return value is equal to AdsErrorCode.NoError the call was successfull.
	TryReadWrite(Int32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32.) [▶ 744]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 745]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32.) [▶ 746]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 747]	Writes data synchronously to an ADS device and then Reads data from this device.

	Name	Description
	TryWrite(Int32, AdsStream, Int32, Int32) [▶ 749]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, .Byte., Int32, Int32) [▶ 750]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 751]	Writes data synchronously to an ADS device.
	TryWriteControl(StateInfo) [▶ 752]	Changes the ADS status and the device status of an ADS server.
	TryWriteControl(StateInfo, AdsStream, Int32, Int32) [▶ 753]	Changes the ADS status and the device status of an ADS server.
	Write(Int32, Int32) [▶ 754]	Trigger Client Method/Command.
	Write(Int32, AdsStream) [▶ 755]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32) [▶ 756]	Trigger Client Method/Command.
	Write(Int32, Int32, AdsStream) [▶ 757]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream) [▶ 758]	Writes data synchronously to an ADS device.
	Write(Int32, AdsStream, Int32, Int32) [▶ 758]	Writes data synchronously to an ADS device.
	Write(Int32, Int32, .Byte., Int32, Int32) [▶ 759]	Writes data synchronously to an ADS device.
	Write(Int32, Int32, AdsStream, Int32, Int32) [▶ 760]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, .Byte., Int32, Int32) [▶ 761]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 762]	Writes data synchronously to an ADS device.
	WriteAny(Int32, Object) [▶ 763]	Writes an object synchronously to an ADS device.
	WriteAny(Int32, Object, .Int32.) [▶ 764]	Writes an object synchronously to an ADS device.

	Name	Description
	WriteAny(UInt32, UInt32, Object) [▶ 765]	Writes an object synchronously to an ADS device.
	WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 766]	Writes an object synchronously to an ADS device.
	WriteAnyString(Int32, String, Int32, Encoding) [▶ 767]	Writes the string (Potentially unsafe!)
	WriteAnyString(UInt32, UInt32, String, Int32, Encoding) [▶ 769]	Writes the string (Potentially unsafe!)
	WriteControl(StateInfo) [▶ 770]	Changes the ADS status and the device status of an ADS server.
	WriteControl(StateInfo, AdsStream, Int32, Int32) [▶ 771]	Changes the ADS status and the device status of an ADS server.
	WriteSymbol(ITcAdsSymbol, Object) [▶ 772]	Writes a value to the symbol. Strings and all primitive data types(UInt32, Int32, Bool etc.) are supported. Array and structures are not supported. If a string is passed as parameter, the method attempts to parse the string according to the ADS data type of the symbol.
	WriteSymbol(String, Object, Boolean) [▶ 773]	Writes the passed object value to the specified ADS symbol.The parameter type must have the same layout as the ADS symbol.

Extension Methods

	Name	Description
	PollAdsState(IObservable.Unit.) [▶ 825]	Overloaded. Gets an observable sequence of AdsState [▶ 365]s via Polling. (Defined by AdsClientExtensions [▶ 821].)
	PollAdsState(TimeSpan) [▶ 826]	Overloaded. Gets an observable sequence of AdsState [▶ 365]s via Polling. (Defined by AdsClientExtensions [▶ 821].)
	PollValues(String, Type, IObservable.Unit.) [▶ 849]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, TimeSpan) [▶ 850]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., TimeSpan) [▶ 853]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)

	Name	Description
	PollValues(String, Type, IObservable.Unit., Func.Exception, Object.) [▶ 854]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, TimeSpan, Func.Exception, Object.) [▶ 855]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., IObservable.Unit., Func.Exception, Object.) [▶ 856]	Overloaded. Polls the symbol values on time points where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues(String, Type, .Int32., TimeSpan, Func.Exception, Object.) [▶ 858]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, IObservable.Unit.) [▶ 842]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, TimeSpan) [▶ 843]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, IObservable.Unit., Func.Exception, T.) [▶ 847]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, TimeSpan, Func.Exception, T.) [▶ 848]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, .Int32., IObservable.Unit.) [▶ 844]	Overloaded. Polls the symbol values on time points where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, .Int32., TimeSpan) [▶ 845]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, .Int32., IObservable.Unit., Func.Exception, T.) [▶ 851]	Overloaded. Polls the symbol values on timepoints where the polling observable streams data / triggers (Defined by AnyTypeExtensions [▶ 835].)
	PollValues.T.(String, .Int32., TimeSpan, Func.Exception, T.) [▶ 852]	Overloaded. Polls the symbol as value sequence of object values with a specified period time. (Defined by AnyTypeExtensions [▶ 835].)

	Name	Description
	WhenAdsStateChanges [▶ 828]	Gets an observable sequence of AdsState [▶ 365]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbol) [▶ 829]	Overloaded. Gets an observable sequence of Notification [▶ 862]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbolCollection) [▶ 830]	Overloaded. Gets an observable sequence of Notification [▶ 862] objects. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbol, NotificationSettings) [▶ 832]	Overloaded. Gets an observable sequence of Notification [▶ 862]s. (Defined by AdsClientExtensions [▶ 821].)
	WhenNotification(ISymbolCollection, NotificationSettings) [▶ 833]	Overloaded. Gets an observable sequence of Notification [▶ 862] objects. (Defined by AdsClientExtensions [▶ 821].)
	WhenValueChanged [▶ 886]	Observable sequence of Values driven by ADS Notifications on the specified symbol. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues.T.(String, IObservable.T.) [▶ 859]	Overloaded. Writes the sequence of values to the symbol specified by the instance path. (Defined by AnyTypeExtensions [▶ 835].)
	WriteValues.T.(String, IObservable.T., Action.Exception.) [▶ 860]	Overloaded. Writes the sequence of values to the symbol specified by the instance path. (Defined by AnyTypeExtensions [▶ 835].)

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.1 TcAdsClient.AddDeviceNotification Method

Overload List

	Name	Description
	AddDeviceNotification(String, AdsStream,	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AdsTransMode, Int32, Int32, Object) [▶ 659]	
	AddDeviceNotification(String, AdsStream, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 660]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 661]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 663]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 665]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 667]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 669]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 671]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.AddDeviceNotification Method (String, AdsStream, AdsTransMode, Int32, Int32, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotification(
    string variableName,
    AdsStream dataStream,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData
)
```

VB

```
Public Function AddDeviceNotification (
    variableName As String,
    dataStream As AdsStream,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.

maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotification(String, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 500]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotification Overload [▶ 657]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AddDeviceNotification Method (String, AdsStream, AdsTransMode, TimeSpan, TimeSpan, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotification(
    string variableName,
    AdsStream dataStream,
    AdsTransMode transMode,
    TimeSpan cycleTime,
    TimeSpan maxDelay,
    Object userData
)
```

VB

```
Public Function AddDeviceNotification (
    variableName As String,
    dataStream As AdsStream,
    transMode As AdsTransMode,
```

```
    cycleTime As TimeSpan,
    maxDelay As TimeSpan,
    userData As Object
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.TimeSpan The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.TimeSpan The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotification Overload [▶ 657]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AddDeviceNotification Method (UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotification(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData
)
```

VB

```
Public Function AddDeviceNotification (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the created ADS notification.

Implements

IAdsNotifications.AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 501]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Examples

The following sample shows how to register/unregister for AdsNotifications (asynchronous change messages) via AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) and DeleteDeviceNotification(Int32) [▶ 697]

Receive AdsNotifications

```
AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event handler
    client.AdsNotification += Client_AdsNotification;

    // Connect to target
    client.Connect("1.2.3.4.5.6", 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a DINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotification("MAIN.nCounter", readStream, AdsTransMode.
OnChange, 200, 0, null);
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotification -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationEventArgs e)
{
    int offset = (int)e.DataStream.Position;
    int length = (int)e.DataStream.Length;

    e.DataStream.Position = offset;
    AdsBinaryReader reader = new AdsBinaryReader(e.DataStream);

    // Read the Unmarshalled data
    //byte[] data = reader.ReadBytes(length);

    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = reader.ReadUInt32();
}
```

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotification Overload [▶ 657]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AddDeviceNotification Method (UInt32, UInt32, AdsStream, AdsTransMode, TimeSpan, TimeSpan, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotification(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    AdsTransMode transMode,
    TimeSpan cycleTime,
    TimeSpan maxDelay,
    Object userData
)
```

VB

```
Public Function AddDeviceNotification (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    transMode As AdsTransMode,
    cycleTime As TimeSpan,
    maxDelay As TimeSpan,
    userData As Object
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.TimeSpan The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.TimeSpan The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Examples

The following sample shows how to register/unregister for AdsNotifications (asynchronous change messages) via AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 667] and DeleteDeviceNotification(Int32) [▶ 697]

Receive AdsNotifications

```
AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event handler
    client.AdsNotification += Client_AdsNotification;

    // Connect to target
    client.Connect("1.2.3.4.5.6", 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a DINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotification("MAIN.nCounter", readStream, AdsTransMode.
OnChange, 200, 0, null);
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotification -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationEventArgs e)
{
    int offset = (int)e.DataStream.Position;
    int length = (int)e.DataStream.Length;

    e.DataStream.Position = offset;
    AdsBinaryReader reader = new AdsBinaryReader(e.DataStream);

    // Read the Unmarshalled data
    //byte[] data = reader.ReadBytes(length);

    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = reader.ReadUInt32();
}
```

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotification Overload [▶ 657]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AddDeviceNotification Method (String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotification(
    string variableName,
    AdsStream dataStream,
    int offset,
    int length,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData
)
```

VB

```
Public Function AddDeviceNotification (
    variableName As String,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 502]

Exceptions

Exception	Condition
ObjectDisposedException	
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Remarks

AdsTransMode [▶ 401]	Parameter semantic
CyclicInContext [▶ 401]	Value of parameter is interpreted as task context number ContextMask [▶ 555]
OnChangeInContext [▶ 401]	Value of parameter is interpreted as task context number ContextMask [▶ 555]

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotification Overload [▶ 657]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

AddDeviceNotificationEx Overload [▶ 673]

TcAdsClient.AddDeviceNotification Method (String, AdsStream, Int32, Int32, AdsTransMode, TimeSpan, TimeSpan, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotification(
    string variableName,
    AdsStream dataStream,
    int offset,
    int length,
    AdsTransMode transMode,
    TimeSpan cycleTime,
    TimeSpan maxDelay,
    Object userData
)
```

VB

```
Public Function AddDeviceNotification (
    variableName As String,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    transMode As AdsTransMode,
    cycleTime As TimeSpan,
    maxDelay As TimeSpan,
    userData As Object
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.TimeSpan The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.TimeSpan The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Remarks

AdsTransMode [▶ 401]	Parameter semantic
CyclicInContext [▶ 401]	Value of parameter is interpreted as task context number ContextMask [▶ 555]
OnChangeInContext [▶ 401]	Value of parameter is interpreted as task context number ContextMask [▶ 555]

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotification Overload [▶ 657]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AddDeviceNotification Method (UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotification(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData
)
```

VB

```
Public Function AddDeviceNotification (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
offset	Type: System.Int32 Byte Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream (in bytes)
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotification(UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, Int32, Int32, Object) [▶ 504]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Examples

The following sample shows how to register/unregister for AdsNotifications (asynchronous change messages) via AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 661] and DeleteDeviceNotification(Int32) [▶ 697]

Receive AdsNotifications

```
AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event handler
    client.AdsNotification += Client_AdsNotification;

    // Connect to target
    client.Connect("1.2.3.4.5.6", 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a DINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotification("MAIN.nCounter", readStream, AdsTransMode.
OnChange, 200, 0, null);
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotification -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationEventArgs e)
{
    int offset = (int)e.DataStream.Position;
    int length = (int)e.DataStream.Length;

    e.DataStream.Position = offset;
    AdsBinaryReader reader = new AdsBinaryReader(e.DataStream);

    // Read the Unmarshalled data
    //byte[] data = reader.ReadBytes(length);

    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = reader.ReadUInt32();
}
```

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotification Overload [▶ 657]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AddDeviceNotification Method (UInt32, UInt32, AdsStream, Int32, Int32, AdsTransMode, TimeSpan, TimeSpan, Object)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotification(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length,
    AdsTransMode transMode,
    TimeSpan cycleTime,
    TimeSpan maxDelay,
    Object userData
)
```

VB

```
Public Function AddDeviceNotification (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    transMode As AdsTransMode,
    cycleTime As TimeSpan,
    maxDelay As TimeSpan,
    userData As Object
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.TimeSpan The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.TimeSpan The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Examples

The following sample shows how to register/unregister for AdsNotifications (asynchronous change messages) via AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 661] and DeleteDeviceNotification(Int32) [▶ 697]

Receive AdsNotifications

```
AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event handler
    client.AdsNotification += Client_AdsNotification;

    // Connect to target
    client.Connect("1.2.3.4.5.6", 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a DINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotification("MAIN.nCounter", readStream, AdsTransMode.
OnChange, 200, 0, null);
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotification -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationEventArgs e)
{
    int offset = (int)e.DataStream.Position;
    int length = (int)e.DataStream.Length;

    e.DataStream.Position = offset;
    AdsBinaryReader reader = new AdsBinaryReader(e.DataStream);
```

```
    // Read the Unmarshalled data
    //byte[] data = reader.ReadBytes(length);

    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = reader.ReadUInt32();
}
```

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotification Overload [▶ 657]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

5.2.61.3.2 TcAdsClient.AddDeviceNotificationEx Method

Overload List

	Name	Description
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type) [▶ 674]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, TimeSpan, TimeSpan, Object, Type) [▶ 676]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 678]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(String, AdsTransMode, TimeSpan, TimeSpan, Object, Type, .Int32.) [▶ 680]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32,	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

	Name	Description
	AdsTransMode, Int32, Int32, Object, Type) [▶ 682]	
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, TimeSpan, TimeSpan, Object, Type) [▶ 683]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 685]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.
	AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, TimeSpan, TimeSpan, Object, Type, .Int32.) [▶ 687]	Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.AddDeviceNotificationEx Method (String, AdsTransMode, Int32, Int32, Object, Type)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotificationEx(
    string variableName,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData,
    Type type
)
```

VB

```
Public Function AddDeviceNotificationEx (
    variableName As String,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
```

```
    userData As Object,
    type As Type
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type) [▶ 506]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Examples

The following sample shows how to use AdsNotificationEx [▶ 776] events.

Receive AdsNotifications

```
//AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event 'Ex' handler
    client.AdsNotificationEx += Client_AdsNotification;

    // Connect to target
    client.Connect(AmsNetId.Local, 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a ZDINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotificationEx("MAIN.nCounter",AdsTransMode.OnChange, 2
00, 0, null,typeof(uint));
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
```

```
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotificationEx -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationExEventArgs e)
{
    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = (uint)e.Value;
}
```

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotificationEx Overload [▶ 673]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AddDeviceNotificationEx Method (String, AdsTransMode, TimeSpan, TimeSpan, Object, Type)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotificationEx(
    string variableName,
    AdsTransMode transMode,
    TimeSpan cycleTime,
    TimeSpan maxDelay,
    Object userData,
    Type type
)
```

VB

```
Public Function AddDeviceNotificationEx (
    variableName As String,
    transMode As AdsTransMode,
    cycleTime As TimeSpan,
    maxDelay As TimeSpan,
    userData As Object,
    type As Type
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.

transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.TimeSpan The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.TimeSpan The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Examples

The following sample shows how to use AdsNotificationEx [▶ 776] events.

Receive AdsNotifications

```
//AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event 'Ex' handler
    client.AdsNotificationEx += Client_AdsNotification;

    // Connect to target
    client.Connect(AmsNetId.Local, 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a ZDINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotificationEx("MAIN.nCounter",AdsTransMode.OnChange, 2
00, 0, null,typeof(uint));
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotificationEx -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationExEventArgs e)
{
    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = (uint)e.Value;
}
```

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotificationEx Overload [▶ 673]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AddDeviceNotificationEx Method (String, AdsTransMode, Int32, Int32, Object, Type, .Int32.)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotificationEx(
    string variableName,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData,
    Type type,
    int[] args
)
```

VB

```
Public Function AddDeviceNotificationEx (
    variableName As String,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object,
    type As Type,
    args As Integer()
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.

type	Type: System.Type Type of the object stored in the event argument.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotificationEx(String, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 507]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Examples

The following sample shows how to use AdsNotificationEx [▶ 776] events.

Receive AdsNotifications

```
//AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event 'Ex' handler
    client.AdsNotificationEx += Client_AdsNotification;

    // Connect to target
    client.Connect(AmsNetId.Local, 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a ZDINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotificationEx("MAIN.nCounter",AdsTransMode.OnChange, 2
00, 0, null,typeof(uint));
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotificationEx -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationExEventArgs e)
{
    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = (uint)e.Value;
}
```

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotificationEx Overload [▶ 673]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AddDeviceNotificationEx Method (String, AdsTransMode, TimeSpan, TimeSpan, Object, Type, .Int32.)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotificationEx(
    string variableName,
    AdsTransMode transMode,
    TimeSpan cycleTime,
    TimeSpan maxDelay,
    Object userData,
    Type type,
    int[] args
)
```

VB

```
Public Function AddDeviceNotificationEx (
    variableName As String,
    transMode As AdsTransMode,
    cycleTime As TimeSpan,
    maxDelay As TimeSpan,
    userData As Object,
    type As Type,
    args As Integer()
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.TimeSpan The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.TimeSpan The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Examples

The following sample shows how to use AdsNotificationEx [▶ 776] events.

Receive AdsNotifications

```
//AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event 'Ex' handler
    client.AdsNotificationEx += Client_AdsNotification;

    // Connect to target
    client.Connect(AmsNetId.Local, 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a ZDINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotificationEx("MAIN.nCounter",AdsTransMode.OnChange, 2
00, 0, null,typeof(uint));
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotificationEx -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationExEventArgs e)
{
    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = (uint)e.Value;
}
```

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotificationEx Overload [▶ 673]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AddDeviceNotificationEx Method (UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotificationEx(
    uint indexGroup,
    uint indexOffset,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
    Object userData,
    Type type
)
```

VB

```
Public Function AddDeviceNotificationEx (
    indexGroup As UInteger,
    indexOffset As UInteger,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object,
    type As Type
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type) [▶ 508]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Examples

The following sample shows how to use AdsNotificationEx [▶ 776] events.

Receive AdsNotifications

```
//AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event 'Ex' handler
    client.AdsNotificationEx += Client_AdsNotification;

    // Connect to target
    client.Connect(AmsNetId.Local, 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a ZDINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotificationEx("MAIN.nCounter",AdsTransMode.OnChange, 2
00, 0, null,typeof(uint));
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotificationEx -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationExEventArgs e)
{
    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = (uint)e.Value;
}
```

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotificationEx Overload [▶ 673]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AddDeviceNotificationEx Method (UInt32, UInt32, AdsTransMode, TimeSpan, TimeSpan, Object, Type)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotificationEx(
    uint indexGroup,
    uint indexOffset,
    AdsTransMode transMode,
    TimeSpan cycleTime,
    TimeSpan maxDelay,
    Object userData,
    Type type
)
```

VB

```
Public Function AddDeviceNotificationEx (
    indexGroup As UInteger,
    indexOffset As UInteger,
    transMode As AdsTransMode,
    cycleTime As TimeSpan,
    maxDelay As TimeSpan,
    userData As Object,
    type As Type
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.TimeSpan The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.TimeSpan The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Examples

The following sample shows how to use AdsNotificationEx [▶ 776] events.

Receive AdsNotifications

```
//AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event 'Ex' handler
    client.AdsNotificationEx += Client_AdsNotification;

    // Connect to target
    client.Connect(AmsNetId.Local, 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a ZDINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotificationEx("MAIN.nCounter",AdsTransMode.OnChange, 2
00, 0, null,typeof(uint));
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotificationEx -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationExEventArgs e)
{
    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = (uint)e.Value;
}
```

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotificationEx Overload [▶ 673]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AddDeviceNotificationEx Method (UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotificationEx(
    uint indexGroup,
    uint indexOffset,
    AdsTransMode transMode,
    int cycleTime,
    int maxDelay,
```

```
    Object userData,
    Type type,
    int[] args
)
```

VB

```
Public Function AddDeviceNotificationEx (
    indexGroup As UInteger,
    indexOffset As UInteger,
    transMode As AdsTransMode,
    cycleTime As Integer,
    maxDelay As Integer,
    userData As Object,
    type As Type,
    args As Integer()
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.Int32 The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.Int32 The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Int32
The handle of the notification.

Implements

IAdsNotifications.AddDeviceNotificationEx(UInt32, UInt32, AdsTransMode, Int32, Int32, Object, Type, .Int32.) [▶ 509]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Examples

The following sample shows how to use AdsNotificationEx [▶ 776] events.

Receive AdsNotifications

```
//AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event 'Ex' handler
    client.AdsNotificationEx += Client_AdsNotification;

    // Connect to target
    client.Connect(AmsNetId.Local, 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a ZDINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotificationEx("MAIN.nCounter",AdsTransMode.OnChange, 2
00, 0, null,typeof(uint));
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotificationEx -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationExEventArgs e)
{
    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = (uint)e.Value;
}
```

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotificationEx Overload [▶ 673]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AddDeviceNotificationEx Method (UInt32, UInt32, AdsTransMode, TimeSpan, TimeSpan, Object, Type, .Int32.)

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event. If type is a string type, the first element of the parameter args specifies the number of characters of the string. If type is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AddDeviceNotificationEx(
    uint indexGroup,
    uint indexOffset,
```

```
    AdsTransMode transMode,
    TimeSpan cycleTime,
    TimeSpan maxDelay,
    Object userData,
    Type type,
    int[] args
)
```

VB

```
Public Function AddDeviceNotificationEx (
    indexGroup As UInteger,
    indexOffset As UInteger,
    transMode As AdsTransMode,
    cycleTime As TimeSpan,
    maxDelay As TimeSpan,
    userData As Object,
    type As Type,
    args As Integer()
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
transMode	Type: TwinCAT.Ads.AdsTransMode [▶ 401] Specifies if the event should be fired cyclically or only if the variable has changed.
cycleTime	Type: System.TimeSpan The ADS server checks whether the variable has changed after this time interval. Unit is in ms.
maxDelay	Type: System.TimeSpan The AdsNotification event is fired at the latest when this time has elapsed. The unit is ms.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Int32
The handle of the notification.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

AddDeviceNotificationEx Overload [▶ 673]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 661]

5.2.61.3.3 TcAdsClient.Close Method

Closes this IConnection [▶ 46]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Close()
```

VB

```
Public Sub Close
```

Implements

IConnection.Close. [▶ 50]

Remarks

The TcAdsClient [▶ 625] is disposed afterwards.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.4 TcAdsClient.Connect Method

Overload List

	Name	Description
	Connect(Int32) [▶ 690]	Establishes a connection to a ADS device using the local netID.
	Connect(AmsAddress) [▶ 690]	Establishes a connection to a ADS device.
	Connect(.Byte., Int32) [▶ 691]	Establishes a connection to a ADS device.
	Connect(String, Int32) [▶ 691]	Establishes a connection to a ADS device.
	Connect(AmsNetId, Int32) [▶ 692]	Establishes a connection to a ADS device.
	Connect(AmsNetId, AmsPort) [▶ 693]	Establishes a connection to a ADS device.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Connect Method (Int32)

Establishes a connection to a ADS device using the local netID.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Connect(
    int srvPort
)
```

VB

```
Public Sub Connect (
    srvPort As Integer
)
```

Parameters

srvPort	Type: System.Int32 Port number of the ADS server.

Reference

TcAdsClient Class [▶ 625]

Connect Overload [▶ 689]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Connect Method (AmsAddress)

Establishes a connection to a ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Connect(
    AmsAddress address
)
```

VB

```
Public Sub Connect (
    address As AmsAddress
)
```

Parameters

address	Type: TwinCAT.Ads.AmsAddress [▶ 410] The address.

Reference

TcAdsClient Class [▶ 625]

Connect Overload [▶ 689]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Connect Method (.Byte., Int32)

Establishes a connection to a ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Connect(
    byte[] netID,
    int srvPort
)
```

VB

```
Public Sub Connect (
    netID As Byte(),
    srvPort As Integer
)
```

Parameters

netID	Type: .System.Byte. NetId of the ADS server.
srvPort	Type: System.Int32 Port number of the ADS server.

Reference

TcAdsClient Class [▶ 625]

Connect Overload [▶ 689]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Connect Method (String, Int32)

Establishes a connection to a ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Connect(
    string netID,
    int srvPort
)
```

VB

```
Public Sub Connect (
    netID As String,
    srvPort As Integer
)
```

Parameters

netID	Type: System.String NetId of the ADS server.
srvPort	Type: System.Int32 Port number of the ADS server.

Reference

TcAdsClient Class [▶ 625]

Connect Overload [▶ 689]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Connect Method (AmsNetId, Int32)

Establishes a connection to a ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Connect(
    AmsNetId netID,
    int srvPort
)
```

VB

```
Public Sub Connect (
    netID As AmsNetId,
    srvPort As Integer
)
```

Parameters

netID	Type: TwinCAT.Ads.AmsNetId [▶ 428] NetId of the ADS server.
srvPort	Type: System.Int32 Port number of the ADS server.

Reference

TcAdsClient Class [▶ 625]

Connect Overload [▶ 689]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Connect Method (AmsNetId, AmsPort)

Establishes a connection to a ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Connect(
    AmsNetId netID,
    AmsPort srvPort
)
```

VB

```
Public Sub Connect (
    netID As AmsNetId,
    srvPort As AmsPort
)
```

Parameters

netID	Type: TwinCAT.Ads.AmsNetId [▶ 428] NetId of the ADS server.
srvPort	Type: TwinCAT.Ads.AmsPort [▶ 450] Port number of the ADS server.

Reference

TcAdsClient Class [▶ 625]

Connect Overload [▶ 689]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.5 TcAdsClient.CreateSymbolInfoLoader Method

Obsolete
This API is now obsolete.

Creates a new instance of the TcAdsSymbolInfoLoader class (Symbol Browser V1, obsolete).

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[ObsoleteAttribute("For new code use the SymbolLoaderFactory.Create method instead!",
    false)]
public TcAdsSymbolInfoLoader CreateSymbolInfoLoader()
```

VB

```
<ObsoleteAttribute("For new code use the SymbolLoaderFactory.Create method instead!",
    false)>
Public Function CreateSymbolInfoLoader As TcAdsSymbolInfoLoader
```

Return Value

Type: TcAdsSymbolInfoLoader [▶ 812]
Instance of the TcAdsSymbolInfoLoader class.

Remarks

This is the traditional way of accessing symbol information from the target device that is still supported here for backward compatibility. For new implementations please consider to use the new symbol browsing capabilities accessed by the SymbolLoaderFactory [▶ 1100] class (Create(IConnection, ISymbolLoaderSettings) [▶ 1102] method).

Examples

Create SymbolLoader V1 object

```
using System;
using System.Diagnostics;
using TwinCAT.Ads;
using TwinCAT.TypeSystem;

namespace Sample
{
    class SymbolBrowserProgramV1
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        ConsoleLogger logger = new ConsoleLogger();
        //logger.Active = false;

        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for Start:");
        Console.ReadLine();

        Stopwatch stopper = new Stopwatch();

        //Parse the AmsAddress from command-line arguments
        AmsAddress address = ArgParser.Parse(args);
        stopper.Start();

        // Create the ADS Client
        using (TcAdsClient client = new TcAdsClient())
        {
        // Establish Connection
        client.Connect(address);

        // Create Symbol / DataType Loader
#pragma warning disable 0618
        TcAdsSymbolInfoLoader loader = client.CreateSymbolInfoLoader();
#pragma warning restore 0618

        // Determine List of used DataTypes.
        ReadOnlyTcAdsDataTypeCollection dataTypes = loader.GetDataTypes(false);

        // Determine List of contained symbols
```

```
        TcAdsSymbolInfoCollection symbols = loader.GetSymbols(false);

        // Dump Datatypes from Target Device
        Console.WriteLine(string.Format("Dumping '{0}' DataTypes:", dataTypes.Count));

        // Iterate through types and Dump Content
        foreach (ITcAdsDataType dataType in dataTypes)
        {
            logger.DumpType(dataType);
        }

        // Dump Symbols from target device
        Console.WriteLine("Dumping '{0}' Symbols:", symbols.Count);

        // Iterates through symbols and Dump Content
        foreach (ITcAdsSymbol5 symbol in symbols)
        {
            logger.DumpSymbol(symbol,0);
        }

        stopper.Stop();
        TimeSpan elapsed = stopper.Elapsed;
        Console.WriteLine("");
        Console.WriteLine("Browsing complete tree: {0},
({1} DataTypes, {2} Symbols)", elapsed, logger.DataTypesCount, logger.SymbolsCount);

        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
        }
    }
    }
}
```

Examples

The following sample shows how to call (Remote Procedures / Methods) within the PLC directly from the TcAdsClient [▶ 625] class.

RPC Call Example

```
namespace Sample
{
    using System;
    using System.Diagnostics;
    using TwinCAT.Ads;
    using TwinCAT.TypeSystem;

    class RpcCallV1Program
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        //Parse the AmsAddress from command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        // Create the ADS Client
        using (TcAdsClient client = new TcAdsClient())
        {
        // Establish Connection
        client.Connect(address);

        // Call a Method that has the following signature (within MAIN Program)

        /*  {attribute 'TcRpcEnable'}
            METHOD PUBLIC M_Add : INT
            VAR_INPUT
            i1 : INT := 0;
            i2 : INT := 0;
            END_VAR
        */

        short result = (short)client.InvokeRpcMethod("MAIN", "M_Add", new object[] {(short)1, (short
)4});

        // Call a Method that has no parameter and returns VOID
```

```
        client.InvokeRpcMethod("MAIN", "M_Method1", new object[] {});
        }
    }
    }
}
```

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.6 TcAdsClient.CreateVariableHandle Method

Generates a unique handle for an ADS variable.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int CreateVariableHandle(
    string variableName
)
```

VB

```
Public Function CreateVariableHandle (
    variableName As String
) As Integer
```

Parameters

variableName	Type: System.String Name of the ADS variable

Return Value

Type: Int32
The handle of the ADS Variable.

Implements

IAdsHandleAccess.CreateVariableHandle(String) [▶ 486]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.7 TcAdsClient.DeleteDeviceNotification Method

Deletes an existing notification.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void DeleteDeviceNotification(
    int notificationHandle
)
```

VB

```
Public Sub DeleteDeviceNotification (
    notificationHandle As Integer
)
```

Parameters

notificationHandle	Type: System.Int32 Handle of the notification.

Implements

IAdsNotifications.DeleteDeviceNotification(Int32) [▶ 510]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Examples

The following sample shows how to register/unregister for AdsNotifications (asynchronous change messages) via AddDeviceNotification(String, AdsStream, Int32, Int32, AdsTransMode, TimeSpan, TimeSpan, Object) [▶ 667] and DeleteDeviceNotification(Int32)

Receive AdsNotifications

```
AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event handler
    client.AdsNotification += Client_AdsNotification;

    // Connect to target
    client.Connect("1.2.3.4.5.6", 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a DINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotification("MAIN.nCounter", readStream, AdsTransMode.
OnChange, 200, 0, null);
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
```

```
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotification -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationEventArgs e)
{
    int offset = (int)e.DataStream.Position;
    int length = (int)e.DataStream.Length;

    e.DataStream.Position = offset;
    AdsBinaryReader reader = new AdsBinaryReader(e.DataStream);

    // Read the Unmarshalled data
    //byte[] data = reader.ReadBytes(length);

    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = reader.ReadUInt32();
}
```

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.8 TcAdsClient.DeleteVariableHandle Method

Releases the handle of a ADS variable again.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void DeleteVariableHandle(
    int variableHandle
)
```

VB

```
Public Sub DeleteVariableHandle (
    variableHandle As Integer
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable

Implements

IAdsHandleAccess.DeleteVariableHandle(Int32) [▶ 487]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.9 TcAdsClient.Disconnect Method

Disconnects the TcAdsClient [▶ 625]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Disconnect()
```

VB

```
Public Function Disconnect As Boolean
```

Return Value

Type: Boolean
true if successfully disconnected, false if the TcAdsClient [▶ 625] was already disconnected.

Implements

IConnection.Disconnect. [▶ 51]

Remarks

The TcAdsClient [▶ 625] can be connected again afterwards. Disconnection doesn't mean disposing / closing:

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.10 TcAdsClient.Dispose Method

Overload List

	Name	Description
	Dispose. [▶ 700]	Releases the resources used by TcAdsClient.
	Dispose(Boolean) [▶ 700]	Disposes the TcAdsClient [▶ 625].

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Dispose Method

Releases the resources used by TcAdsClient.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Dispose()
```

VB

```
Public Sub Dispose
```

Implements

IDisposable.Dispose.

Reference

TcAdsClient Class [▶ 625]

Dispose Overload [▶ 699]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Dispose Method (Boolean)

Disposes the TcAdsClient [▶ 625].

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual void Dispose(
    bool disposing
)
```

VB

```
Protected Overridable Sub Dispose (
    disposing As Boolean
)
```

Parameters

disposing	Type: System.Boolean

Remarks

When overwritten don't forget to call the base class

Reference

TcAdsClient Class [▶ 625]

Dispose Overload [▶ 699]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.11 TcAdsClient.Finalize Method

Finalizes an instance of the TcAdsClient [▶ 625] class.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected override void Finalize()
```

VB

```
Protected Overrides Sub Finalize
```

Implements

Object.Finalize.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.12 TcAdsClient.InvokeRpcMethod Method

Overload List

	Name	Description
	InvokeRpcMethod(String, Int32, .Object.) [▶ 702]	Invokes the specified RPC Method
	InvokeRpcMethod(String, String, .Object.) [▶ 703]	Invokes the RPC method.
	InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 705]	Invokes the specified RPC Method

	Name	Description
	InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 706]	Invokes the RPC method.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.InvokeRpcMethod Method (String, Int32, .Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object InvokeRpcMethod(
    string symbolPath,
    int methodId,
    Object[] parameters
)
```

VB

```
Public Function InvokeRpcMethod (
    symbolPath As String,
    methodId As Integer,
    parameters As Object()
) As Object
```

Parameters

symbolPath	Type: System.String The symbol path.
methodId	Type: System.Int32 The method identifier.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
System.Object.

Implements

ITcAdsRpcInvoke.InvokeRpcMethod(String, Int32, .Object.) [▶ 537]

Examples

The following sample shows how to call (Remote Procedures / Methods) within the PLC.

Dynamic Tree Mode

```
namespace Sample
{
    using System;
    using System.Diagnostics;
    using TwinCAT.Ads;
    using TwinCAT.TypeSystem;

    class RpcCallV1Program
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        //Parse the AmsAddress from command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        // Create the ADS Client
        using (TcAdsClient client = new TcAdsClient())
        {
        // Establish Connection
        client.Connect(address);

        // Call a Method that has the following signature (within MAIN Program)

        /*  {attribute 'TcRpcEnable'}
            METHOD PUBLIC M_Add : INT
            VAR_INPUT
            i1 : INT := 0;
            i2 : INT := 0;
            END_VAR
        */

        short result = (short)client.InvokeRpcMethod("MAIN", "M_Add", new object[] {(short)1, (short
)4});

        // Call a Method that has no parameter and returns VOID
        client.InvokeRpcMethod("MAIN", "M_Method1", new object[] {});
        }
    }
    }
}
```

Reference

TcAdsClient Class [▶ 625]

InvokeRpcMethod Overload [▶ 701]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.InvokeRpcMethod Method (String, String, .Object.)

Invokes the RPC method.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object InvokeRpcMethod(
    string symbolPath,
    string methodName,
    Object[] parameters
)
```

VB

```
Public Function InvokeRpcMethod (
    symbolPath As String,
    methodName As String,
    parameters As Object()
) As Object
```

Parameters

symbolPath	Type: System.String The symbol path.
methodName	Type: System.String Name of the method.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
System.Object.

Implements

ITcAdsRpcInvoke.InvokeRpcMethod(String, String, .Object.) [▶ 538]

Exceptions

Exception	Condition
NotImplementedException	

Examples

The following sample shows how to call (Remote Procedures / Methods) within the PLC.

Dynamic Tree Mode

```
namespace Sample
{
    using System;
    using System.Diagnostics;
    using TwinCAT.Ads;
    using TwinCAT.TypeSystem;

    class RpcCallV1Program
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        //Parse the AmsAddress from command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        // Create the ADS Client
        using (TcAdsClient client = new TcAdsClient())
        {
        // Establish Connection
        client.Connect(address);

        // Call a Method that has the following signature (within MAIN Program)

        /*  {attribute 'TcRpcEnable'}
            METHOD PUBLIC M_Add : INT
```

```
            VAR_INPUT
            i1 : INT := 0;
            i2 : INT := 0;
            END_VAR
        */

        short result = (short)client.InvokeRpcMethod("MAIN", "M_Add", new object[] {(short)1, (short
)4});

        // Call a Method that has no parameter and returns VOID
        client.InvokeRpcMethod("MAIN", "M_Method1", new object[] {});
        }
    }
    }
}
```

Reference

TcAdsClient Class [▶ 625]

InvokeRpcMethod Overload [▶ 701]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.InvokeRpcMethod Method (ITcAdsSymbol, Int32, .Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object InvokeRpcMethod(
    ITcAdsSymbol symbol,
    int methodId,
    Object[] parameters
)
```

VB

```
Public Function InvokeRpcMethod (
    symbol As ITcAdsSymbol,
    methodId As Integer,
    parameters As Object()
) As Object
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
methodId	Type: System.Int32 The method identifier / Virtual Function table index.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
System.Object.

Implements

ITcAdsRpcInvoke.InvokeRpcMethod(ITcAdsSymbol, Int32, .Object.) [▶ 539]

Examples

The following sample shows how to call (Remote Procedures / Methods) within the PLC.

Dynamic Tree Mode

```
namespace Sample
{
    using System;
    using System.Diagnostics;
    using TwinCAT.Ads;
    using TwinCAT.TypeSystem;

    class RpcCallV1Program
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        //Parse the AmsAddress from command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        // Create the ADS Client
        using (TcAdsClient client = new TcAdsClient())
        {
        // Establish Connection
        client.Connect(address);

        // Call a Method that has the following signature (within MAIN Program)

        /*  {attribute 'TcRpcEnable'}
            METHOD PUBLIC M_Add : INT
            VAR_INPUT
            i1 : INT := 0;
            i2 : INT := 0;
            END_VAR
        */

        short result = (short)client.InvokeRpcMethod("MAIN", "M_Add", new object[] {(short)1, (short
)4});

        // Call a Method that has no parameter and returns VOID
        client.InvokeRpcMethod("MAIN", "M_Method1", new object[] {});
        }
    }
    }
}
```

Reference

TcAdsClient Class [▶ 625]

InvokeRpcMethod Overload [▶ 701]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.InvokeRpcMethod Method (ITcAdsSymbol, String, .Object.)

Invokes the RPC method.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object InvokeRpcMethod(
    ITcAdsSymbol symbol,
    string methodName,
    Object[] parameters
)
```

VB

```
Public Function InvokeRpcMethod (
    symbol As ITcAdsSymbol,
    methodName As String,
    parameters As Object()
) As Object
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
methodName	Type: System.String Name of the method.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
System.Object.

Implements

ITcAdsRpcInvoke.InvokeRpcMethod(ITcAdsSymbol, String, .Object.) [▶ 540]

Exceptions

Exception	Condition
NotImplementedException	

Reference

TcAdsClient Class [▶ 625]

InvokeRpcMethod Overload [▶ 701]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.13 TcAdsClient.OnBeforeDisconnect Method

Called when before the TcAdsClient [▶ 625] is disconnected.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual void OnBeforeDisconnect()
```

VB

```
Protected Overridable Sub OnBeforeDisconnect
```

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.14 TcAdsClient.OnConnectionStateChanged Method

Called when the ConnectionState of the TcAdsClient [▶ 625] has changed.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual void OnConnectionStateChanged(
    ConnectionState newState,
    ConnectionState oldState
)
```

VB

```
Protected Overridable Sub OnConnectionStateChanged (
    newState As ConnectionState,
    oldState As ConnectionState
)
```

Parameters

newState	Type: TwinCAT.ConnectionState [▶ 39] The new state.
oldState	Type: TwinCAT.ConnectionState [▶ 39] The old state.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.15 TcAdsClient.Read Method

Overload List

	Name	Description
	Read(Int32, AdsStream) [▶ 709]	Reads data synchronously from an ADS device and writes it to the given stream.

	Name	Description
	Read(UInt32, UInt32, AdsStream) [▶ 710]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(Int32, AdsStream, Int32, Int32) [▶ 711]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, .Byte., Int32, Int32) [▶ 712]	Reads data synchronously from an ADS device and writes it to the given stream.
	Read(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 713]	Reads data synchronously from an ADS device and writes it to the given stream.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Read Method (Int32, AdsStream)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Read(
    int variableHandle,
    AdsStream dataStream
)
```

VB

```
Public Function Read (
    variableHandle As Integer,
    dataStream As AdsStream
) As Integer
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.

Return Value

Type: Int32
Number of successfully returned data bytes.

Implements

IAdsHandleAccess.Read(Int32, AdsStream) [▶ 488]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Read Overload [▶ 708]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Read Method (UInt32, UInt32, AdsStream)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Read(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream
)
```

VB

```
Public Function Read (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.

Return Value

Type: Int32
Number of successfully returned data bytes.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Read Overload [▶ 708]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Read Method (Int32, AdsStream, Int32, Int32)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Read(
    int variableHandle,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Function Read (
    variableHandle As Integer,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
) As Integer
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Implements

IAdsHandleAccess.Read(Int32, AdsStream, Int32, Int32) [▶ 489]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Read Overload [▶ 708]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Read Method (UInt32, UInt32, .Byte., Int32, Int32)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Read(
    uint indexGroup,
    uint indexOffset,
    byte[] dataStream,
    int offset,
    int length
)
```

VB

```
Public Function Read (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As Byte(),
    offset As Integer,
    length As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: .System.Byte. Stream that receives the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Read Overload [▶ 708]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Read Method (UInt32, UInt32, AdsStream, Int32, Int32)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Read(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Function Read (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Read Overload [▶ 708]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.16 TcAdsClient.ReadAny Method

Overload List

	Name	Description
	ReadAny(Int32, Type) [▶ 714]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(Int32, Type, .Int32.) [▶ 715]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type) [▶ 716]	Reads data synchronously from an ADS device and writes it to an object.
	ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 717]	Reads data synchronously from an ADS device and writes it to an object.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.ReadAny Method (Int32, Type)

Reads data synchronously from an ADS device and writes it to an object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadAny(
    int variableHandle,
    Type type
)
```

VB

```
Public Function ReadAny (
    variableHandle As Integer,
    type As Type
) As Object
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable.
type	Type: System.Type Type of the object to be read.

Return Value

Type: Object
The object the read data is written to.

Implements

IAdsAnyAccess.ReadAny(Int32, Type) [▶ 461]

Reference

TcAdsClient Class [▶ 625]

ReadAny Overload [▶ 714]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.ReadAny Method (Int32, Type, .Int32.)

Reads data synchronously from an ADS device and writes it to an object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadAny(
    int variableHandle,
    Type type,
    int[] args
)
```

VB

```
Public Function ReadAny (
    variableHandle As Integer,
    type As Type,
    args As Integer()
) As Object
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable.
type	Type: System.Type Type of the object to be read.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Object
The object the read data is written to.

Implements

IAdsAnyAccess.ReadAny(Int32, Type, .Int32.) [▶ 461]

Remarks

If the Type of the object to be read is a string type, the first element of the parameter args specifies the number of characters of the string. If the Type of the object to be read is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Reference

TcAdsClient Class [▶ 625]

ReadAny Overload [▶ 714]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.ReadAny Method (UInt32, UInt32, Type)

Reads data synchronously from an ADS device and writes it to an object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadAny(
    uint indexGroup,
    uint indexOffset,
    Type type
)
```

VB

```
Public Function ReadAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    type As Type
) As Object
```

Parameters

indexGroup	Type: System.UInt32 Index group of the ADS variable.
indexOffset	Type: System.UInt32 Index offset of the ADS variable.
type	Type: System.Type Type of the object to be read.

Return Value

Type: Object
The object the read data is written to.

Implements

IAdsAnyAccess.ReadAny(UInt32, UInt32, Type) [▶ 462]

Examples

Usage of ReadAny/WriteAny

```
using (TcAdsClient client = new TcAdsClient())
{
    UInt32 valueToRead = 0;
    UInt32 valueToWrite = 42;

    client.Connect(AmsNetId.Local, 851);
    adsClient.WriteAny(0x4020, 0x0, valueToWrite);
    valueToRead = (uint)adsClient.ReadAny(0x4020, 0x0, typeof(UInt32));
}
```

Reference

TcAdsClient Class [▶ 625]

ReadAny Overload [▶ 714]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.ReadAny Method (UInt32, UInt32, Type, .Int32.)

Reads data synchronously from an ADS device and writes it to an object.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadAny(
    uint indexGroup,
    uint indexOffset,
    Type type,
    int[] args
)
```

VB

```
Public Function ReadAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    type As Type,
    args As Integer()
) As Object
```

Parameters

indexGroup	Type: System.UInt32 Index group of the ADS variable.
indexOffset	Type: System.UInt32 Index offset of the ADS variable.
type	Type: System.Type Type of the object to be read.
args	Type: .System.Int32. Additional arguments.

Return Value

Type: Object
The object the read data is written to.

Implements

IAdsAnyAccess.ReadAny(UInt32, UInt32, Type, .Int32.) [▶ 463]

Remarks

If the Type of the object to be read is a string type, the first element of the parameter args specifies the number of characters of the string. If the Type of the object to be read is an array type, the number of elements for each dimension has to be specified in the parameter args. At the moment only 1 dimensional Arrays are supported.

Reference

TcAdsClient Class [▶ 625]

ReadAny Overload [▶ 714]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.17 TcAdsClient.ReadAnyString Method

Overload List

	Name	Description
	ReadAnyString(Int32, Int32, Encoding) [▶ 718]	Reads the string.
	ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 720]	Reads the string

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.ReadAnyString Method (Int32, Int32, Encoding)

Reads the string.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ReadAnyString(
    int variableHandle,
    int len,
    Encoding encoding
)
```

VB

```
Public Function ReadAnyString (
    variableHandle As Integer,
    len As Integer,
    encoding As Encoding
) As String
```

Parameters

variableHandle	Type: System.Int32 The variable handle.
len	Type: System.Int32 The length of the string (e.g. 80 for STRING[80] or WSTRING[80]
encoding	Type: System.Text.Encoding The encoding (Default or Unicode).

Return Value

Type: String
System.String.

Implements

IAdsAnyAccess.ReadAnyString(Int32, Int32, Encoding) [▶ 464]

Examples

The following code shows how to Read/Write string values with the ANY concept.

Read/Write Any Strings

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local port 851 (PLC)

    int stringHandle = client.CreateVariableHandle("MAIN.string"); // Symbol "string" defined in MAI
N as STRING
    int wStringHandle = client.CreateVariableHandle("MAIN.wstring"); // Symbol "string" defined in M
AIN as WSTRING

    try
    {
    string str = client.ReadAnyString(stringHandle, 80, Encoding.Default);
    string wStr = client.ReadAnyString(wStringHandle, 80, Encoding.Unicode);

    string changedValue = "Changed";

    // Attention, take care that the memory of the string in the process image is not exceeded!
    client.WriteAnyString(stringHandle, changedValue, 80, Encoding.Default);
    client.WriteAnyString(wStringHandle, changedValue, 80, Encoding.Unicode);
    }
    finally
    {
    client.DeleteVariableHandle(stringHandle);
    client.DeleteVariableHandle(wStringHandle);
    }
}
```

Reference

TcAdsClient Class [▶ 625]

ReadAnyString Overload [▶ 718]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.ReadAnyString Method (UInt32, UInt32, Int32, Encoding)

Reads the string

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ReadAnyString(
    uint indexGroup,
    uint indexOffset,
    int len,
    Encoding encoding
)
```

VB

```
Public Function ReadAnyString (
    indexGroup As UInteger,
    indexOffset As UInteger,
    len As Integer,
    encoding As Encoding
) As String
```

Parameters

indexGroup	Type: System.UInt32 The index group.
indexOffset	Type: System.UInt32 The index offset.
len	Type: System.Int32 The length of the string.
encoding	Type: System.Text.Encoding The encoding (Default or Unicode).

Return Value

Type: String
System.String.

Implements

IAdsAnyAccess.ReadAnyString(UInt32, UInt32, Int32, Encoding) [▶ 465]

Examples

The following code shows how to Read/Write string values with the ANY concept.

Read/Write Any Strings

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local port 851 (PLC)

    int stringHandle = client.CreateVariableHandle("MAIN.string"); // Symbol "string" defined in MAI
N as STRING
    int wStringHandle = client.CreateVariableHandle("MAIN.wstring"); // Symbol "string" defined in M
AIN as WSTRING

    try
    {
    string str = client.ReadAnyString(stringHandle, 80, Encoding.Default);
```

```
    string wStr = client.ReadAnyString(wStringHandle, 80, Encoding.Unicode);

    string changedValue = "Changed";

    // Attention, take care that the memory of the string in the process image is not exceeded!
    client.WriteAnyString(stringHandle, changedValue, 80, Encoding.Default);
    client.WriteAnyString(wStringHandle, changedValue, 80, Encoding.Unicode);
    }
    finally
    {
    client.DeleteVariableHandle(stringHandle);
    client.DeleteVariableHandle(wStringHandle);
    }
}
```

Reference

TcAdsClient Class [▶ 625]

ReadAnyString Overload [▶ 718]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.18 TcAdsClient.ReadDeviceInfo Method

Reads the identification and version number of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DeviceInfo ReadDeviceInfo()
```

VB

```
Public Function ReadDeviceInfo As DeviceInfo
```

Return Value

Type: DeviceInfo [▶ 455]
DeviceInfo struct containing the name of the device and the version information.

Implements

IAdsConnection.ReadDeviceInfo. [▶ 483]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.19 TcAdsClient.ReadState Method

Reads the ADS status and the device status from an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public StateInfo ReadState()
```

VB

```
Public Function ReadState As StateInfo
```

Return Value

Type: StateInfo [▶ 598]
The ADS statue and device status.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.20 TcAdsClient.ReadSymbol Method

Overload List

	Name	Description
	ReadSymbol(ITcAdsSymbol) [▶ 722]	Reads the value of a symbol and returns it as an object. Strings and all primitive datatypes(UInt32, Int32, Bool etc.) are supported. Arrays and structures cannot be read.
	ReadSymbol(String, Type, Boolean) [▶ 723]	Reads the value of a symbol and returns the value as object. The parameter type must have the same layout as the ADS symbol.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.ReadSymbol Method (ITcAdsSymbol)

Reads the value of a symbol and returns it as an object. Strings and all primitive datatypes(UInt32, Int32, Bool etc.) are supported. Arrays and structures cannot be read.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadSymbol(
    ITcAdsSymbol symbol
)
```

VB

```
Public Function ReadSymbol (
    symbol As ITcAdsSymbol
) As Object
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol that should be read.

Return Value

Type: Object
The value of the symbol as an object.

Exceptions

Exception	Condition
AdsDatatypeNotSupportedException [▶ 300]	Thrown when a ADS datatype is not supported.
AdsErrorException [▶ 310]	Thrown when the Sync port is not open.

Reference

TcAdsClient Class [▶ 625]

ReadSymbol Overload [▶ 722]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.ReadSymbol Method (String, Type, Boolean)

Reads the value of a symbol and returns the value as object. The parameter type must have the same layout as the ADS symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadSymbol(
    string name,
    Type type,
    bool reloadSymbolInfo
)
```

VB

```
Public Function ReadSymbol (
    name As String,
    type As Type,
    reloadSymbolInfo As Boolean
) As Object
```

Parameters

name	Type: System.String Name of the ADS symbol.
type	Type: System.Type Managed type of the ADS symbol.
reloadSymbolInfo	Type: System.Boolean If reload is true previously stored symbol information is cleared. As a consequence the symbol information must be obtained from the ADS server again.

Return Value

Type: Object
Value of the symbol

Reference

TcAdsClient Class [▶ 625]

ReadSymbol Overload [▶ 722]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.21 TcAdsClient.ReadSymbolInfo Method

Call this method to obtain information about the individual symbols (variables) in ADS devices.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ITcAdsSymbol ReadSymbolInfo(
    string name
)
```

VB

```
Public Function ReadSymbolInfo (
    name As String
) As ITcAdsSymbol
```

Parameters

name	Type: System.String Name of the symbol.

Return Value

Type: ITcAdsSymbol [▶ 549]
A ITcAdsSymbol containing the requested symbol information or null if symbol could not be found.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.22 TcAdsClient.ReadWrite Method

Overload List

	Name	Description
	ReadWrite(UInt32, UInt32, AdsStream, AdsStream) [▶ 725]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 726]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32) [▶ 727]	Writes data synchronously to an ADS device and then Reads data from this device.
	ReadWrite(UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 729]	Writes data synchronously to an ADS device and then Reads data from this device.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.ReadWrite Method (UInt32, UInt32, AdsStream, AdsStream)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReadWrite(
    uint indexGroup,
    uint indexOffset,
    AdsStream rdDataStream,
    AdsStream wrDataStream
)
```

VB

```
Public Function ReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    rdDataStream As AdsStream,
    wrDataStream As AdsStream
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.

Return Value

Type: Int32
Number of successfully returned data bytes.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

ReadWrite Overload [▶ 725]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.ReadWrite Method (Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReadWrite(
    int variableHandle,
    AdsStream rdDataStream,
    int rdOffset,
    int rdLength,
    AdsStream wrDataStream,
    int wrOffset,
    int wrLength
)
```

VB

```
Public Function ReadWrite (
    variableHandle As Integer,
    rdDataStream As AdsStream,
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As AdsStream,
    wrOffset As Integer,
    wrLength As Integer
) As Integer
```

Parameters

variableHandle	Type: System.Int32 Variable handle.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Implements

IAdsHandleAccess.ReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32) [▶ 490]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

ReadWrite Overload [▶ 725]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.ReadWrite Method (UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReadWrite(
    uint indexGroup,
    uint indexOffset,
    byte[] rdDataStream,
    int rdOffset,
    int rdLength,
    byte[] wrDataStream,
    int wrOffset,
    int wrLength
)
```

VB

```
Public Function ReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    rdDataStream As Byte(),
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As Byte(),
    wrOffset As Integer,
    wrLength As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
rdDataStream	Type: .System.Byte. Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: .System.Byte. Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

ReadWrite Overload [▶ 725]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.ReadWrite Method (UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReadWrite(
    uint indexGroup,
    uint indexOffset,
    AdsStream rdDataStream,
    int rdOffset,
    int rdLength,
    AdsStream wrDataStream,
    int wrOffset,
    int wrLength
)
```

VB

```
Public Function ReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    rdDataStream As AdsStream,
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As AdsStream,
    wrOffset As Integer,
    wrLength As Integer
) As Integer
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.

Return Value

Type: Int32
Number of successfully returned data bytes.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

ReadWrite Overload [▶ 725]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.23 TcAdsClient.TryAddDeviceNotification Method

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryAddDeviceNotification(
    string variableName,
    AdsStream dataStream,
    int offset,
    int length,
    NotificationSettings settings,
    Object userData,
    out uint handle
)
```

VB

```
Public Function TryAddDeviceNotification (
    variableName As String,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    settings As NotificationSettings,
    userData As Object,
    <OutAttribute> ByRef handle As UInteger
) As AdsErrorCode
```

Parameters

variableName	Type: System.String Name of the ADS variable.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] The stream that should receive the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
settings	Type: TwinCAT.Ads.NotificationSettings [▶ 573] The notification settings.
userData	Type: System.Object This object can be used to store user specific data.
handle	Type: System.UInt32. The handle.

Return Value

Type: AdsErrorCode [▶ 305]
The handle of the notification.

Implements

IAdsNotifications.TryAddDeviceNotification(String, AdsStream, Int32, Int32, NotificationSettings, Object, UInt32.) [▶ 511]

Exceptions

Exception	Condition
ObjectDisposedException	
AdsErrorException [▶ 310]	

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

AddDeviceNotificationEx Overload [▶ 673]

5.2.61.3.24 TcAdsClient.TryAddDeviceNotificationEx Method

Connects a variable to the ADS client. The ADS client will be notified by the AdsNotification event.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryAddDeviceNotificationEx(
    string variableName,
    NotificationSettings settings,
    Object userData,
    Type type,
    int[] args,
    out uint handle
)
```

VB

```
Public Function TryAddDeviceNotificationEx (
    variableName As String,
    settings As NotificationSettings,
    userData As Object,
    type As Type,
    args As Integer(),
    <OutAttribute> ByRef handle As UInteger
) As AdsErrorCode
```

Parameters

variableName	Type: System.String Name of the ADS variable.
settings	Type: TwinCAT.Ads.NotificationSettings [▶ 573] The settings.
userData	Type: System.Object This object can be used to store user specific data.
type	Type: System.Type Type of the object stored in the event argument.
args	Type: .System.Int32. Additional arguments.
handle	Type: System.UInt32. The handle.

Return Value

Type: AdsErrorCode [▶ 305]
The handle of the notification.

Implements

IAdsNotifications.TryAddDeviceNotificationEx(String, NotificationSettings, Object, Type, .Int32., UInt32.) [▶ 512]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.25 TcAdsClient.TryDeleteDeviceNotification Method

Deletes an existing notification.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryDeleteDeviceNotification(
    uint notificationHandle
)
```

VB

```
Public Function TryDeleteDeviceNotification (
    notificationHandle As UInteger
) As AdsErrorCode
```

Parameters

notificationHandle	Type: System.UInt32 Handle of the notification.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Implements

IAdsNotifications.TryDeleteDeviceNotification(UInt32) [▶ 513]

Exceptions

Exception	Condition
ObjectDisposedException	

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.26 TcAdsClient.TryInvokeRpcMethod Method

Overload List

	Name	Description
	TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 734]	Invokes the specified RPC Method
	TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 735]	Tries to invoke the RPC method.
	TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 736]	Invokes the specified RPC Method
	TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 737]	Tries to invoke a RPC method.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryInvokeRpcMethod Method (String, Int32, .Object., Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryInvokeRpcMethod(
    string symbolPath,
    int methodId,
    Object[] parameters,
    out Object retValue
)
```

VB

```
Public Function TryInvokeRpcMethod (
    symbolPath As String,
    methodId As Integer,
    parameters As Object(),
    <OutAttribute> ByRef retValue As Object
) As AdsErrorCode
```

Parameters

symbolPath	Type: System.String The symbol path.
methodId	Type: System.Int32 The method identifier.
parameters	Type: .System.Object. The parameters.
retValue	Type: System.Object. The return value.

Return Value

Type: AdsErrorCode [▶ 305]
true if succeeded, false otherwise.

Implements

ITcAdsRpcInvoke.TryInvokeRpcMethod(String, Int32, .Object., Object.) [▶ 541]

Exceptions

Exception	Condition
ArgumentOutOfRangeException	symbolPath or methodId
ArgumentNullException	parameters
RpcMethodNotSupportedException [▶ 586]	

Reference

TcAdsClient Class [▶ 625]

TryInvokeRpcMethod Overload [▶ 733]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryInvokeRpcMethod Method (String, String, .Object., Object.)

Tries to invoke the RPC method.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryInvokeRpcMethod(
    string symbolPath,
    string methodName,
    Object[] parameters,
    out Object returnValue
)
```

VB

```
Public Function TryInvokeRpcMethod (
    symbolPath As String,
    methodName As String,
    parameters As Object(),
    <OutAttribute> ByRef returnValue As Object
) As AdsErrorCode
```

Parameters

symbolPath	Type: System.String The symbol path.
methodName	Type: System.String Name of the method.
parameters	Type: .System.Object. The parameters.
returnValue	Type: System.Object. The return value.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Implements

ITcAdsRpcInvoke.TryInvokeRpcMethod(String, String, .Object., Object.) [▶ 542]

Exceptions

Exception	Condition
NotImplementedException	

Reference

TcAdsClient Class [▶ 625]

TryInvokeRpcMethod Overload [▶ 733]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryInvokeRpcMethod Method (ITcAdsSymbol, Int32, .Object., Object.)

Invokes the specified RPC Method

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryInvokeRpcMethod(
    ITcAdsSymbol symbol,
    int methodId,
    Object[] parameters,
    out Object retValue
)
```

VB

```
Public Function TryInvokeRpcMethod (
    symbol As ITcAdsSymbol,
    methodId As Integer,
    parameters As Object(),
    <OutAttribute> ByRef retValue As Object
) As AdsErrorCode
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
methodId	Type: System.Int32 The method identifier / Virtual Function table index.
parameters	Type: .System.Object. The parameters.
retValue	Type: System.Object. The return value.

Return Value

Type: AdsErrorCode [▶ 305]
true if succeeded, false otherwise.

Implements

ITcAdsRpcInvoke.TryInvokeRpcMethod(ITcAdsSymbol, Int32, .Object., Object.) [▶ 543]

Exceptions

Exception	Condition
ArgumentNullException	symbol or parameters
ArgumentOutOfRangeException	methodId
RpcMethodNotSupportedException [▶ 586]	

Reference

TcAdsClient Class [▶ 625]

TryInvokeRpcMethod Overload [▶ 733]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryInvokeRpcMethod Method (ITcAdsSymbol, String, .Object., Object.)

Tries to invoke a RPC method.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryInvokeRpcMethod(
    ITcAdsSymbol symbol,
    string methodName,
    Object[] parameters,
    out Object returnValue
)
```

VB

```
Public Function TryInvokeRpcMethod (
    symbol As ITcAdsSymbol,
    methodName As String,
    parameters As Object(),
    <OutAttribute> ByRef returnValue As Object
) As AdsErrorCode
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol.
methodName	Type: System.String Name of the method.
parameters	Type: .System.Object. The parameters.
returnValue	Type: System.Object. The return value.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Implements

ITcAdsRpcInvoke.TryInvokeRpcMethod(ITcAdsSymbol, String, .Object., Object.) [▶ 544]

Exceptions

Exception	Condition
ArgumentNullException	symbol or parameters
ArgumentOutOfRangeException	methodName
AdsSymbolException [▶ 389]	

Reference

TcAdsClient Class [▶ 625]

TryInvokeRpcMethod Overload [▶ 733]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.27 TcAdsClient.TryRead Method

Overload List

	Name	Description
	TryRead(UInt32, UInt32, AdsStream, Int32.) [▶ 739]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(Int32, AdsStream, Int32, Int32, Int32.) [▶ 739]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, .Byte., Int32, Int32, Int32.) [▶ 740]	Reads data synchronously from an ADS device and writes it to the given stream.
	TryRead(UInt32, UInt32, AdsStream, Int32, Int32, Int32.) [▶ 741]	Reads data synchronously from an ADS device and writes it to the given stream.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryRead Method (UInt32, UInt32, AdsStream, Int32.)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryRead(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    out int readBytes
)
```

VB

```
Public Function TryRead (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
readBytes	Type: System.Int32. Number of successfully returned data bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

TcAdsClient Class [▶ 625]

TryRead Overload [▶ 738]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryRead Method (Int32, AdsStream, Int32, Int32, Int32.)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryRead(
    int variableHandle,
    AdsStream dataStream,
    int offset,
    int length,
    out int readBytes
)
```

VB

```
Public Function TryRead (
    variableHandle As Integer,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
readBytes	Type: System.Int32. Number of successfully returned data bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode

Implements

IAdsHandleAccess.TryRead(Int32, AdsStream, Int32, Int32, Int32.) [▶ 491]

Exceptions

Exception	Condition
ArgumentException	

Reference

TcAdsClient Class [▶ 625]

TryRead Overload [▶ 738]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryRead Method (UInt32, UInt32, .Byte., Int32, Int32, Int32.)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryRead(
    uint indexGroup,
    uint indexOffset,
    byte[] dataStream,
    int offset,
    int length,
    out int readBytes
)
```

VB

```
Public Function TryRead (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As Byte(),
    offset As Integer,
    length As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: .System.Byte. Stream that receives the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
readBytes	Type: System.Int32. Number of successfully returned data bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

TcAdsClient Class [▶ 625]

TryRead Overload [▶ 738]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryRead Method (UInt32, UInt32, AdsStream, Int32, Int32, Int32.)

Reads data synchronously from an ADS device and writes it to the given stream.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryRead(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length,
    out int readBytes
)
```

VB

```
Public Function TryRead (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.
readBytes	Type: System.Int32. Number of successfully returned data bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

TcAdsClient Class [▶ 625]

TryRead Overload [▶ 738]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.28 TcAdsClient.TryReadState Method

Reads the ADS status and the device status from an ADS server. Unlike the ReadState method this method does not call an exception on failure. Instead an AdsErrorCode is returned. If the return value is equal to AdsErrorCode.NoError the call was successfull.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReadState(
    out StateInfo stateInfo
)
```

VB

```
Public Function TryReadState (
    <OutAttribute> ByRef stateInfo As StateInfo
) As AdsErrorCode
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598]. The ADS statue and device status.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode of the ads read state call. Check for AdsErrorCode.NoError to see if call was successfull.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.29 TcAdsClient.TryReadWrite Method

Overload List

	Name	Description
	TryReadWrite(Int32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32.) [▶ 744]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 745]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32.) [▶ 746]	Writes data synchronously to an ADS device and then Reads data from this device.
	TryReadWrite(UInt32, UInt32, AdsStream, Int32,	Writes data synchronously to an ADS device and then Reads data from this device.

	Name	Description
	Int32, AdsStream, Int32, Int32, Int32.) [▶ 747]	

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryReadWrite Method (Int32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32.)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReadWrite(
    int variableHandle,
    byte[] rdDataStream,
    int rdOffset,
    int rdLength,
    byte[] wrDataStream,
    int wrOffset,
    int wrLength,
    out int readBytes
)
```

VB

```
Public Function TryReadWrite (
    variableHandle As Integer,
    rdDataStream As Byte(),
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As Byte(),
    wrOffset As Integer,
    wrLength As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

variableHandle	Type: System.Int32 Variable handle.
rdDataStream	Type: .System.Byte. Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: .System.Byte. Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.

wrLength	Type: System.Int32 Length of the data in wrDataStream.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ArgumentException	

Reference

TcAdsClient Class [▶ 625]

TryReadWrite Overload [▶ 743]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryReadWrite Method (Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReadWrite(
    int variableHandle,
    AdsStream rdDataStream,
    int rdOffset,
    int rdLength,
    AdsStream wrDataStream,
    int wrOffset,
    int wrLength,
    out int readBytes
)
```

VB

```
Public Function TryReadWrite (
    variableHandle As Integer,
    rdDataStream As AdsStream,
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As AdsStream,
    wrOffset As Integer,
    wrLength As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

variableHandle	Type: System.Int32 Variable handle.

rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Implements

IAdsHandleAccess.TryReadWrite(Int32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.) [▶ 492]

Exceptions

Exception	Condition
ArgumentException	

Reference

TcAdsClient Class [▶ 625]

TryReadWrite Overload [▶ 743]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryReadWrite Method (UInt32, UInt32, .Byte., Int32, Int32, .Byte., Int32, Int32, Int32.)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReadWrite(
    uint indexGroup,
    uint indexOffset,
    byte[] rdDataStream,
    int rdOffset,
    int rdLength,
    byte[] wrDataStream,
    int wrOffset,
```

```
    int wrLength,
    out int readBytes
)
```

VB

```
Public Function TryReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    rdDataStream As Byte(),
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As Byte(),
    wrOffset As Integer,
    wrLength As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
rdDataStream	Type: .System.Byte. Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: .System.Byte. Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

TcAdsClient Class [▶ 625]

TryReadWrite Overload [▶ 743]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryReadWrite Method (UInt32, UInt32, AdsStream, Int32, Int32, AdsStream, Int32, Int32, Int32.)

Writes data synchronously to an ADS device and then Reads data from this device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReadWrite(
    uint indexGroup,
    uint indexOffset,
    AdsStream rdDataStream,
    int rdOffset,
    int rdLength,
    AdsStream wrDataStream,
    int wrOffset,
    int wrLength,
    out int readBytes
)
```

VB

```
Public Function TryReadWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    rdDataStream As AdsStream,
    rdOffset As Integer,
    rdLength As Integer,
    wrDataStream As AdsStream,
    wrOffset As Integer,
    wrLength As Integer,
    <OutAttribute> ByRef readBytes As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
rdDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data that has been read.
rdOffset	Type: System.Int32 Offset of the data in rdDataStream.
rdLength	Type: System.Int32 Length of the data in rdDataStream.
wrDataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be written.
wrOffset	Type: System.Int32 Offset of the data in wrDataStream.
wrLength	Type: System.Int32 Length of the data in wrDataStream.
readBytes	Type: System.Int32. The read bytes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

TcAdsClient Class [▶ 625]

TryReadWrite Overload [▶ 743]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.30 TcAdsClient.TryWrite Method

Overload List

	Name	Description
	TryWrite(Int32, AdsStream, Int32, Int32) [▶ 749]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, .Byte., Int32, Int32) [▶ 750]	Writes data synchronously to an ADS device.
	TryWrite(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 751]	Writes data synchronously to an ADS device.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryWrite Method (Int32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWrite(
    int variableHandle,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Function TryWrite (
    variableHandle As Integer,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
) As AdsErrorCode
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Implements

IAdsHandleAccess.TryWrite(Int32, AdsStream, Int32, Int32) [▶ 493]

Exceptions

Exception	Condition
ArgumentException	

Reference

TcAdsClient Class [▶ 625]

TryWrite Overload [▶ 749]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryWrite Method (UInt32, UInt32, .Byte., Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWrite(
    uint indexGroup,
    uint indexOffset,
    byte[] writeBuffer,
    int offset,
    int length
)
```

VB

```
Public Function TryWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    writeBuffer As Byte(),
    offset As Integer,
    length As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
writeBuffer	Type: .System.Byte. The write buffer.
offset	Type: System.Int32 Offset of the data in dataStream.

length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

TcAdsClient Class [▶ 625]

TryWrite Overload [▶ 749]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryWrite Method (UInt32, UInt32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWrite(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Function TryWrite (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
) As AdsErrorCode
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

TcAdsClient Class [▶ 625]

TryWrite Overload [▶ 749]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.31 TcAdsClient.TryWriteControl Method

Overload List

	Name	Description
	TryWriteControl(StateInfo) [▶ 752]	Changes the ADS status and the device status of an ADS server.
	TryWriteControl(StateInfo, AdsStream, Int32, Int32) [▶ 753]	Changes the ADS status and the device status of an ADS server.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryWriteControl Method (StateInfo)

Changes the ADS status and the device status of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWriteControl(
    StateInfo stateInfo
)
```

VB

```
Public Function TryWriteControl (
    stateInfo As StateInfo
) As AdsErrorCode
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598] New ADS status and device status.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

TcAdsClient Class [▶ 625]

TryWriteControl Overload [▶ 752]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.TryWriteControl Method (StateInfo, AdsStream, Int32, Int32)

Changes the ADS status and the device status of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWriteControl(
    StateInfo stateInfo,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Function TryWriteControl (
    stateInfo As StateInfo,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
) As AdsErrorCode
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598] New ADS status and device status.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be sent to the ADS device
offset	Type: System.Int32 Offset of the data in the stream.
length	Type: System.Int32 Length of the data in the stream.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Exceptions

Exception	Condition
ArgumentException	

Reference

TcAdsClient Class [▶ 625]

TryWriteControl Overload [▶ 752]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.32 TcAdsClient.Write Method

Overload List

	Name	Description
	Write(Int32, Int32) [▶ 754]	Trigger Client Method/Command.
	Write(Int32, AdsStream) [▶ 755]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32) [▶ 756]	Trigger Client Method/Command.
	Write(Int32, Int32, AdsStream) [▶ 757]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream) [▶ 758]	Writes data synchronously to an ADS device.
	Write(Int32, AdsStream, Int32, Int32) [▶ 758]	Writes data synchronously to an ADS device.
	Write(Int32, Int32, .Byte., Int32, Int32) [▶ 759]	Writes data synchronously to an ADS device.
	Write(Int32, Int32, AdsStream, Int32, Int32) [▶ 760]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, .Byte., Int32, Int32) [▶ 761]	Writes data synchronously to an ADS device.
	Write(UInt32, UInt32, AdsStream, Int32, Int32) [▶ 762]	Writes data synchronously to an ADS device.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Write Method (Int32, Int32)

Trigger Client Method/Command.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    int indexGroup,
    int indexOffset
)
```

VB

```
Public Sub Write (
    indexGroup As Integer,
    indexOffset As Integer
)
```

Parameters

indexGroup	Type: System.Int32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.Int32 Contains the index offset number of the requested ADS service.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Remarks

This method is used to trigger Client Methods/Commands without parameters.

Reference

TcAdsClient Class [▶ 625]

Write Overload [▶ 754]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Write Method (Int32, AdsStream)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    int variableHandle,
    AdsStream dataStream
)
```

VB

```
Public Sub Write (
    variableHandle As Integer,
    dataStream As AdsStream
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.

Implements

IAdsHandleAccess.Write(Int32, AdsStream) [▶ 494]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Write Overload [▶ 754]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Write Method (UInt32, UInt32)

Trigger Client Method/Command.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    uint indexGroup,
    uint indexOffset
)
```

VB

```
Public Sub Write (
    indexGroup As UInteger,
    indexOffset As UInteger
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Remarks

This method is used to trigger Client Methods/Commands without parameters.

Reference

TcAdsClient Class [▶ 625]

Write Overload [▶ 754]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Write Method (Int32, Int32, AdsStream)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    int indexGroup,
    int indexOffset,
    AdsStream dataStream
)
```

VB

```
Public Sub Write (
    indexGroup As Integer,
    indexOffset As Integer,
    dataStream As AdsStream
)
```

Parameters

indexGroup	Type: System.Int32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.Int32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Write Overload [▶ 754]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Write Method (UInt32, UInt32, AdsStream)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream
)
```

VB

```
Public Sub Write (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that receives the data.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Write Overload [▶ 754]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Write Method (Int32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    int variableHandle,
    AdsStream dataStream,
```

```
    int offset,
    int length
)
```

VB

```
Public Sub Write (
    variableHandle As Integer,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Implements

IAdsHandleAccess.Write(Int32, AdsStream, Int32, Int32) [▶ 495]

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Write Overload [▶ 754]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Write Method (Int32, Int32, .Byte., Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    int indexGroup,
    int indexOffset,
    byte[] writeBuffer,
    int offset,
    int length
)
```

VB

```
Public Sub Write (
    indexGroup As Integer,
    indexOffset As Integer,
    writeBuffer As Byte(),
    offset As Integer,
    length As Integer
)
```

Parameters

indexGroup	Type: System.Int32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.Int32 Contains the index offset number of the requested ADS service.
writeBuffer	Type: .System.Byte. The write buffer.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Write Overload [▶ 754]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Write Method (Int32, Int32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    int indexGroup,
    int indexOffset,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Sub Write (
    indexGroup As Integer,
    indexOffset As Integer,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
)
```

Parameters

indexGroup	Type: System.Int32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.Int32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Write Overload [▶ 754]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Write Method (UInt32, UInt32, .Byte., Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    uint indexGroup,
    uint indexOffset,
    byte[] dataStream,
    int offset,
    int length
)
```

VB

```
Public Sub Write (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As Byte(),
    offset As Integer,
    length As Integer
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.

dataStream	Type: .System.Byte. Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Write Overload [▶ 754]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.Write Method (UInt32, UInt32, AdsStream, Int32, Int32)

Writes data synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    uint indexGroup,
    uint indexOffset,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Sub Write (
    indexGroup As UInteger,
    indexOffset As UInteger,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data.
offset	Type: System.Int32 Offset of the data in dataStream.
length	Type: System.Int32 Length of the data in dataStream.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

Write Overload [▶ 754]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.33 TcAdsClient.WriteAny Method

Overload List

	Name	Description
	WriteAny(Int32, Object) [▶ 763]	Writes an object synchronously to an ADS device.
	WriteAny(Int32, Object, .Int32.) [▶ 764]	Writes an object synchronously to an ADS device.
	WriteAny(UInt32, UInt32, Object) [▶ 765]	Writes an object synchronously to an ADS device.
	WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 766]	Writes an object synchronously to an ADS device.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.WriteAny Method (Int32, Object)

Writes an object synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteAny(
    int variableHandle,
    Object value
)
```

VB

```
Public Sub WriteAny (
    variableHandle As Integer,
    value As Object
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable.
value	Type: System.Object Object to write to the ADS device.

Implements

IAdsAnyAccess.WriteAny(Int32, Object) [▶ 466]

Reference

TcAdsClient Class [▶ 625]

WriteAny Overload [▶ 763]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.WriteAny Method (Int32, Object, .Int32.)

Writes an object synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteAny(
    int variableHandle,
    Object value,
    int[] args
)
```

VB

```
Public Sub WriteAny (
    variableHandle As Integer,
    value As Object,
    args As Integer()
)
```

Parameters

variableHandle	Type: System.Int32 Handle of the ADS variable.
value	Type: System.Object Object to write to the ADS device.
args	Type: .System.Int32. Additional arguments.

Implements

IAdsAnyAccess.WriteAny(Int32, Object, .Int32.) [▶ 467]

Remarks

If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.

Reference

TcAdsClient Class [▶ 625]

WriteAny Overload [▶ 763]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.WriteAny Method (UInt32, UInt32, Object)

Writes an object synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteAny(
    uint indexGroup,
    uint indexOffset,
    Object value
)
```

VB

```
Public Sub WriteAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    value As Object
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
value	Type: System.Object Object to write to the ADS device.

Implements

IAdsAnyAccess.WriteAny(UInt32, UInt32, Object) [▶ 468]

Examples

Usage of ReadAny/WriteAny

```
using (TcAdsClient client = new TcAdsClient())
{
    UInt32 valueToRead = 0;
    UInt32 valueToWrite = 42;

    client.Connect(AmsNetId.Local, 851);
    adsClient.WriteAny(0x4020, 0x0, valueToWrite);
    valueToRead = (uint)adsClient.ReadAny(0x4020, 0x0, typeof(UInt32));
}
```

Reference

TcAdsClient Class [▶ 625]

WriteAny Overload [▶ 763]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.WriteAny Method (UInt32, UInt32, Object, .Int32.)

Writes an object synchronously to an ADS device.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteAny(
    uint indexGroup,
    uint indexOffset,
    Object value,
    int[] args
)
```

VB

```
Public Sub WriteAny (
    indexGroup As UInteger,
    indexOffset As UInteger,
    value As Object,
    args As Integer()
)
```

Parameters

indexGroup	Type: System.UInt32 Contains the index group number of the requested ADS service.
indexOffset	Type: System.UInt32 Contains the index offset number of the requested ADS service.
value	Type: System.Object Object to write to the ADS device.
args	Type: .System.Int32. Additional arguments.

Implements

IAdsAnyAccess.WriteAny(UInt32, UInt32, Object, .Int32.) [▶ 468]

Remarks

If the Type of the object to be written is a string type, the first element of parameter args specifies the number of characters of the string.

Reference

TcAdsClient Class [▶ 625]

WriteAny Overload [▶ 763]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.34 TcAdsClient.WriteAnyString Method

Overload List

	Name	Description
	WriteAnyString(Int32, String, Int32, Encoding) [▶ 767]	Writes the string (Potentially unsafe!)
	WriteAnyString(UInt32, UInt32, String, Int32, Encoding) [▶ 769]	Writes the string (Potentially unsafe!)

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.WriteAnyString Method (Int32, String, Int32, Encoding)

Writes the string (Potentially unsafe!)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteAnyString(
    int variableHandle,
    string value,
    int length,
    Encoding encoding
)
```

VB

```
Public Sub WriteAnyString (
    variableHandle As Integer,
    value As String,
    length As Integer,
    encoding As Encoding
)
```

Parameters

variableHandle	Type: System.Int32 The variable handle.
value	Type: System.String The value.
length	Type: System.Int32 The length of the string to write
encoding	Type: System.Text.Encoding The encoding.

Exceptions

Exception	Condition
NotImplementedException	

Remarks

NOTE

Data can be overwritten

Potentially this method is unsafe because following data can be overwritten after the string symbol. Please be sure to specify the string length lower than the string size reserved within the process image!

Examples

The following code shows how to Read/Write string values with the ANY concept.

Read/Write Any Strings

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local port 851 (PLC)

    int stringHandle = client.CreateVariableHandle("MAIN.string"); // Symbol "string" defined in MAI
N as STRING
    int wStringHandle = client.CreateVariableHandle("MAIN.wstring"); // Symbol "string" defined in M
AIN as WSTRING

    try
    {
    string str = client.ReadAnyString(stringHandle, 80, Encoding.Default);
    string wStr = client.ReadAnyString(wStringHandle, 80, Encoding.Unicode);

    string changedValue = "Changed";

    // Attention, take care that the memory of the string in the process image is not exceeded!
    client.WriteAnyString(stringHandle, changedValue, 80, Encoding.Default);
    client.WriteAnyString(wStringHandle, changedValue, 80, Encoding.Unicode);
    }
    finally
    {
    client.DeleteVariableHandle(stringHandle);
    client.DeleteVariableHandle(wStringHandle);
    }
}
```

Reference

TcAdsClient Class [▶ 625]

WriteAnyString Overload [▶ 767]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.WriteAnyString Method (UInt32, UInt32, String, Int32, Encoding)

Writes the string (Potentially unsafe!)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteAnyString(
    uint indexGroup,
    uint indexOffset,
    string value,
    int length,
    Encoding encoding
)
```

VB

```
Public Sub WriteAnyString (
    indexGroup As UInteger,
    indexOffset As UInteger,
    value As String,
    length As Integer,
    encoding As Encoding
)
```

Parameters

indexGroup	Type: System.UInt32 The index group.
indexOffset	Type: System.UInt32 The index offset.
value	Type: System.String The value.
length	Type: System.Int32 The length.
encoding	Type: System.Text.Encoding The encoding.

Exceptions

Exception	Condition
NotImplementedException	

Remarks

NOTE

Data can be overwritten

Potentially this method is unsafe because following data can be overwritten after the string symbol. Please be sure to specify the string length lower than the string size reserved within the process image!

Examples

The following code shows how to Read/Write string values with the ANY concept.

Read/Write Any Strings

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local port 851 (PLC)

    int stringHandle = client.CreateVariableHandle("MAIN.string"); // Symbol "string" defined in MAI
N as STRING
    int wStringHandle = client.CreateVariableHandle("MAIN.wstring"); // Symbol "string" defined in M
AIN as WSTRING

    try
    {
    string str = client.ReadAnyString(stringHandle, 80, Encoding.Default);
    string wStr = client.ReadAnyString(wStringHandle, 80, Encoding.Unicode);

    string changedValue = "Changed";

    // Attention, take care that the memory of the string in the process image is not exceeded!
    client.WriteAnyString(stringHandle, changedValue, 80, Encoding.Default);
    client.WriteAnyString(wStringHandle, changedValue, 80, Encoding.Unicode);
    }
    finally
    {
    client.DeleteVariableHandle(stringHandle);
    client.DeleteVariableHandle(wStringHandle);
    }
}
```

Reference

TcAdsClient Class [▶ 625]

WriteAnyString Overload [▶ 767]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.35 TcAdsClient.WriteControl Method

Overload List

	Name	Description
	WriteControl(StateInfo) [▶ 770]	Changes the ADS status and the device status of an ADS server.
	WriteControl(StateInfo, AdsStream, Int32, Int32) [▶ 771]	Changes the ADS status and the device status of an ADS server.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.WriteControl Method (StateInfo)

Changes the ADS status and the device status of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteControl(
    StateInfo stateInfo
)
```

VB

```
Public Sub WriteControl (
    stateInfo As StateInfo
)
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598] New ADS status and device status.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

WriteControl Overload [▶ 770]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.WriteControl Method (StateInfo, AdsStream, Int32, Int32)

Changes the ADS status and the device status of an ADS server.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteControl(
    StateInfo stateInfo,
    AdsStream dataStream,
    int offset,
    int length
)
```

VB

```
Public Sub WriteControl (
    stateInfo As StateInfo,
    dataStream As AdsStream,
    offset As Integer,
    length As Integer
)
```

Parameters

stateInfo	Type: TwinCAT.Ads.StateInfo [▶ 598] New ADS status and device status.

dataStream	Type: TwinCAT.Ads.AdsStream [▶ 375] Stream that contains the data that should be sent to the ADS device
offset	Type: System.Int32 Offset of the data in the stream.
length	Type: System.Int32 Length of the data in the stream.

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

WriteControl Overload [▶ 770]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.3.36 TcAdsClient.WriteSymbol Method

Overload List

	Name	Description
	WriteSymbol(ITcAdsSymbol, Object) [▶ 772]	Writes a value to the symbol. Strings and all primitive data types(UInt32, Int32, Bool etc.) are supported. Array and structures are not supported. If a string is passed as parameter, the method attempts to parse the string according to the ADS data type of the symbol.
	WriteSymbol(String, Object, Boolean) [▶ 773]	Writes the passed object value to the specified ADS symbol.The parameter type must have the same layout as the ADS symbol.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.WriteSymbol Method (ITcAdsSymbol, Object)

Writes a value to the symbol. Strings and all primitive data types(UInt32, Int32, Bool etc.) are supported. Array and structures are not supported. If a string is passed as parameter, the method attempts to parse the string according to the ADS data type of the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteSymbol(
    ITcAdsSymbol symbol,
    Object val
)
```

VB

```
Public Sub WriteSymbol (
    symbol As ITcAdsSymbol,
    val As Object
)
```

Parameters

symbol	Type: TwinCAT.Ads.ITcAdsSymbol [▶ 549] The symbol the value is written to.
val	Type: System.Object The value to write.

Exceptions

Exception	Condition
AdsDatatypeNotSupportedException [▶ 300]	Thrown when a ADS data type is not supported.
AdsErrorException [▶ 310]	Thrown when the ADS call fails.

Reference

TcAdsClient Class [▶ 625]

WriteSymbol Overload [▶ 772]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.WriteSymbol Method (String, Object, Boolean)

Writes the passed object value to the specified ADS symbol.The parameter type must have the same layout as the ADS symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteSymbol(
    string name,
    Object value,
    bool reloadSymbolInfo
)
```

VB

```
Public Sub WriteSymbol (
    name As String,
    value As Object,
    reloadSymbolInfo As Boolean
)
```

Parameters

name	Type: System.String Name of the ADS symbol.
value	Type: System.Object Object holding the value to be written to the ADS symbol

reloadSymbolInfo	Type: System.Boolean If reload is true previously stored symbol information is cleared. As a consequence the symbol information must be obtained from the ADS server again.

Reference

TcAdsClient Class [▶ 625]

WriteSymbol Overload [▶ 772]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.4 TcAdsClient Events

The TcAdsClient [▶ 625] type exposes the following members.

Events

	Name	Description
	AdsNotification [▶ 774]	Occurs when the ADS device sends a ADS Notification to the client.
	AdsNotificationError [▶ 776]	Occurs when a exception has occurred during notification management.
	AdsNotificationEx [▶ 776]	Occurs when the ADS devices sends an (extended) notification to the client.
	AdsStateChanged [▶ 778]	Occurs when the ADS state changes.
	AdsSymbolVersionChanged [▶ 778]	Occurs when the symbol version has been changed changes.
	AmsRouterNotification [▶ 779]	Occurs when the state of the local Router has changed.
	ConnectionStateChanged [▶ 779]	Occurs when the connection state has been changed.

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.4.1 TcAdsClient.AdsNotification Event

Occurs when the ADS device sends a ADS Notification to the client.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event AdsNotificationEventHandler AdsNotification
```

VB

```
Public Event AdsNotification As AdsNotificationEventHandler
```

Value

Type: TwinCAT.Ads.AdsNotificationEventHandler [▶ 342]

Implements

IAdsNotifications.AdsNotification [▶ 514]

Remarks

In most implementations, these notifications indicate changed values on the client target. These 'ADS notifications' will be received asynchronously from the target system and distributed via this AdsNotification event. **IMPORTANT: The Default setting of the** Synchronize [▶ 646] **property has changed to 'false' from Version 4.2.XX on. This has the effect that - by default - the notifications events AdsNotification and are not synchronized into the UI thread anymore. To re-enable the obsolete behavior for legacy applications set to 'true'.**AdsNotificationEx [▶ 776]Synchronize [▶ 646]

Examples

The following sample shows how to register/unregister for AdsNotifications (asynchronous change messages) via AddDeviceNotification(UInt32, UInt32, AdsStream, AdsTransMode, Int32, Int32, Object) [▶ 661] and DeleteDeviceNotification(Int32) [▶ 697]

Receive AdsNotifications

```
AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event handler
    client.AdsNotification += Client_AdsNotification;

    // Connect to target
    client.Connect("1.2.3.4.5.6", 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a DINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotification("MAIN.nCounter", readStream, AdsTransMode.
OnChange, 200, 0, null);
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotification -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationEventArgs e)
{
    int offset = (int)e.DataStream.Position;
    int length = (int)e.DataStream.Length;
```

```
    e.DataStream.Position = offset;
    AdsBinaryReader reader = new AdsBinaryReader(e.DataStream);

    // Read the Unmarshalled data
    //byte[] data = reader.ReadBytes(length);

    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = reader.ReadUInt32();
}
```

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotificationEx [▶ 776]

TcAdsClient.AdsNotificationError [▶ 776]

5.2.61.4.2 TcAdsClient.AdsNotificationError Event

Occurs when a exception has occurred during notification management.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event AdsNotificationErrorEventHandler AdsNotificationError
```

VB

```
Public Event AdsNotificationError As AdsNotificationErrorEventHandler
```

Value

Type: TwinCAT.Ads.AdsNotificationErrorEventHandler [▶ 335]

Implements

IAdsNotifications.AdsNotificationError [▶ 515]

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationEx [▶ 776]

5.2.61.4.3 TcAdsClient.AdsNotificationEx Event

Occurs when the ADS devices sends an (extended) notification to the client.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event AdsNotificationExEventHandler AdsNotificationEx
```

VB

```
Public Event AdsNotificationEx As AdsNotificationExEventHandler
```

Value

Type: TwinCAT.Ads.AdsNotificationExEventHandler [▶ 346]

Implements

IAdsNotifications.AdsNotificationEx [▶ 516]

Remarks

In most implementations, these notifications indicate changed values on the client target. These 'ADS notifications' will be received asynchronously from the target system and distributed via this AdsNotification [▶ 774] event. **IMPORTANT: The Default setting of the** Synchronize [▶ 646] **property has changed to 'false' from Version 4.2.XX on. This has the effect that - by default - the notifications events and AdsNotificationEx are not synchronized into the UI thread anymore. To re-enable the obsolete behavior for legacy applications set to 'true'.**AdsNotification [▶ 774]Synchronize [▶ 646]

Examples

The following sample shows how to use AdsNotificationEx events.

Receive AdsNotifications

```
//AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event 'Ex' handler
    client.AdsNotificationEx += Client_AdsNotification;

    // Connect to target
    client.Connect(AmsNetId.Local, 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a ZDINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotificationEx("MAIN.nCounter",AdsTransMode.OnChange, 2
00, 0, null,typeof(uint));
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotificationEx -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationExEventArgs e)
{
```

```
    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = (uint)e.Value;
}
```

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

TcAdsClient.AddDeviceNotificationEx(Int64, Int64, AdsTransMode, Int32, Int32, Object, Type)

TcAdsClient.DeleteDeviceNotification(Int32) [▶ 697]

TcAdsClient.AdsNotification [▶ 774]

TcAdsClient.AdsNotificationError [▶ 776]

5.2.61.4.4 TcAdsClient.AdsStateChanged Event

Occurs when the ADS state changes.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event AdsStateChangedEventHandler AdsStateChanged
```

VB

```
Public Event AdsStateChanged As AdsStateChangedEventHandler
```

Value

Type: TwinCAT.Ads.AdsStateChangedEventHandler [▶ 374]

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.4.5 TcAdsClient.AdsSymbolVersionChanged Event

Occurs when the symbol version has been changed changes.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event EventHandler AdsSymbolVersionChanged
```

VB

```
Public Event AdsSymbolVersionChanged As EventHandler
```

Value

Type: System.EventHandler

Remarks

This is the case when the connected ADS server restarts. This invalidates all actual opened symbol handles. The SymbolVersion counter doesn't trigger, when an online change is made on the PLC (ports 801, ..., 851 ...)

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.4.6 TcAdsClient.AmsRouterNotification Event

Occurs when the state of the local Router has changed.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event AmsRouterNotificationEventHandler AmsRouterNotification
```

VB

```
Public Event AmsRouterNotification As AmsRouterNotificationEventHandler
```

Value

Type: TwinCAT.Ads.AmsRouterNotificationEventHandler [▶ 454]

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.61.4.7 TcAdsClient.ConnectionStateChanged Event

Occurs when the connection state has been changed.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event EventHandler<ConnectionStateChangedEventArgs> ConnectionStateChanged
```

VB

```
Public Event ConnectionStateChanged As EventHandler(Of ConnectionStateChangedEventArgs)
```

Value

Type: System.EventHandler.ConnectionStateChangedEventArgs [▶ 39].

Implements

IConnectionStateProvider.ConnectionStateChanged [▶ 54]

Reference

TcAdsClient Class [▶ 625]

TwinCAT.Ads Namespace [▶ 108]

5.2.62 TcAdsSymbolInfo Class

The class TcAdsSymbolInfo represents a symbol loaded by an instance of the TcAdsSymbolInfoLoader class.

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TcAdsSymbolInfo
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class TcAdsSymbolInfo : ITcAdsSymbol5,
    ITcAdsSymbol4, ITcAdsSymbol3, ITcAdsSymbol2, ITcAdsSymbol, ITcAdsSymbolBrowser
```

VB

```
Public Class TcAdsSymbolInfo
    Implements ITcAdsSymbol5, ITcAdsSymbol4, ITcAdsSymbol3, ITcAdsSymbol2,
    ITcAdsSymbol, ITcAdsSymbolBrowser
```

The TcAdsSymbolInfo type exposes the following members.

Properties

	Name	Description
	ArrayDimensions [▶ 783]	Gets the array dimensions.
	ArrayInfos [▶ 784]	Gets the collection of Array Infos.
	Attributes [▶ 785]	Gets the attributes of the ITcAdsSymbol [▶ 549]
	BitSize [▶ 785]	Gets the size of this TcAdsSymbolInfo in bits.
	ByteSize [▶ 786]	Gets the (aligned) size of of the Type/Instance in Bytes
	Category [▶ 786]	Gets the Data Type Category
	Comment [▶ 787]	Gets the comment behind the variable declaration.

	Name	Description
	ContextMask [▶ 788]	Gets the ContextMask of the symbol, indicating the task the variable belongs to. If ContextMask is not zero use CyclicInContext [▶ 401] or OnChangeInContext [▶ 401] to add notifications.
	DataType [▶ 788]	Gets the DataType object
	DataTypeId [▶ 789]	Data type of the symbol.
	HasRpcMethods [▶ 789]	Gets a value indicating whether this instance has RPC methods (Struct types only)
	IndexGroup [▶ 790]	Gets the index group of the symbol.
	IndexOffset [▶ 790]	Gets the index offset of the symbol.
	IsBitType [▶ 791]	Indicates if the BitValue flag is set for the symbol.
	IsOversamplingArray [▶ 792]	Gets a value indicating whether this instance is oversampling array.
	IsPersistent [▶ 792]	Indicates if the Persistent flag is set for the symbol.
	IsReadOnly [▶ 793]	Indicates if the ReadOnly flag is set for the symbol.
	IsStatic [▶ 793]	Indicates, that this symbol is a static symbol.
	IsTcComInterfacePointer [▶ 794]	Indicates if the TcComInterfacePointer flag is set for the symbol.
	IsTypeGuid [▶ 794]	Indicates if the TypeGuid flag is set for the symbol.
	Name [▶ 795]	Gets the name of the symbol.
	Parent [▶ 796]	Gets the parent of this symbol.
	RpcMethods [▶ 796]	Gets the RPC method descriptions
	ShortName [▶ 797]	Gets the name of the symbol (short form without prefixed names of the parents).
	Size [▶ 797]	Gets the size of the symbol.
	SubSymbols [▶ 798]	Gets the sub symbols of this symbol as a collection of TcAdsSymbolInfo objects.
	TypeName [▶ 798]	Gets the name of the symbol data type.

Methods

	Name	Description
	Equals [▶ 799]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 800]	Gets the HashCode of the Address (Overrides Object.GetHashCode.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)

	Name	Description
	IsRecursive [▶ 801]	Gets a value indicating whether this instance is recursive.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString [▶ 801]	Returns a String that represents this instance. (Overrides Object.ToString..)
	TryGetPointerRef [▶ 802]	Gets the Base data type of the pointer (the referenced type)
	TryGetReference [▶ 802]	Gets the referenced data type of the reference.

Operators

	Name	Description
	Equality [▶ 803]	Implements the == operator.
	Inequality [▶ 804]	Implements the != operator.

Reference

TwinCAT.Ads Namespace [▶ 108]

TwinCAT.Ads.ITcAdsSymbol5 [▶ 566]

TwinCAT.Ads.ITcAdsSymbolBrowser [▶ 572]

5.2.62.1 TcAdsSymbolInfo Properties

The TcAdsSymbolInfo [▶ 780] type exposes the following members.

Properties

	Name	Description
	ArrayDimensions [▶ 783]	Gets the array dimensions.
	ArrayInfos [▶ 784]	Gets the collection of Array Infos.
	Attributes [▶ 785]	Gets the attributes of the ITcAdsSymbol [▶ 549]
	BitSize [▶ 785]	Gets the size of this TcAdsSymbolInfo [▶ 780] in bits.
	ByteSize [▶ 786]	Gets the (aligned) size of of the Type/Instance in Bytes
	Category [▶ 786]	Gets the Data Type Category
	Comment [▶ 787]	Gets the comment behind the variable declaration.
	ContextMask [▶ 788]	Gets the ContextMask of the symbol, indicating the task the variable belongs to. If ContextMask is not zero use CyclicInContext [▶ 401] or OnChangeInContext [▶ 401] to add notifications.
	DataType [▶ 788]	Gets the DataType object

	Name	Description
	DataTypeId [▶ 789]	Data type of the symbol.
	HasRpcMethods [▶ 789]	Gets a value indicating whether this instance has RPC methods (Struct types only)
	IndexGroup [▶ 790]	Gets the index group of the symbol.
	IndexOffset [▶ 790]	Gets the index offset of the symbol.
	IsBitType [▶ 791]	Indicates if the BitValue flag is set for the symbol.
	IsOversamplingArray [▶ 792]	Gets a value indicating whether this instance is oversampling array.
	IsPersistent [▶ 792]	Indicates if the Persistent flag is set for the symbol.
	IsReadOnly [▶ 793]	Indicates if the ReadOnly flag is set for the symbol.
	IsStatic [▶ 793]	Indicates, that this symbol is a static symbol.
	IsTcComInterfacePointer [▶ 794]	Indicates if the TcComInterfacePointer flag is set for the symbol.
	IsTypeGuid [▶ 794]	Indicates if the TypeGuid flag is set for the symbol.
	Name [▶ 795]	Gets the name of the symbol.
	Parent [▶ 796]	Gets the parent of this symbol.
	RpcMethods [▶ 796]	Gets the RPC method descriptions
	ShortName [▶ 797]	Gets the name of the symbol (short form without prefixed names of the parents).
	Size [▶ 797]	Gets the size of the symbol.
	SubSymbols [▶ 798]	Gets the sub symbols of this symbol as a collection of TcAdsSymbolInfo objects.
	TypeName [▶ 798]	Gets the name of the symbol data type.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.1 TcAdsSymbolInfo.ArrayDimensions Property

Gets the array dimensions.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ArrayDimensions { get; }
```

VB

```
Public ReadOnly Property ArrayDimensions As Integer
    Get
```

Property Value

Type: Int32
The array dimensions.

Implements

ITcAdsSymbol3.ArrayDimensions [▶ 560]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.2 TcAdsSymbolInfo.ArrayInfos Property

Gets the collection of Array Infos.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsDatatypeArrayInfo[] ArrayInfos { get; }
```

VB

```
Public ReadOnly Property ArrayInfos As AdsDatatypeArrayInfo()
    Get
```

Property Value

Type: .AdsDatatypeArrayInfo [▶ 296].
The array infos.

Implements

ITcAdsSymbol3.ArrayInfos [▶ 560]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.3 TcAdsSymbolInfo.Attributes Property

Gets the attributes of the ITcAdsSymbol [▶ 549]

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyTypeAttributeCollection Attributes { get; }
```

VB

```
Public ReadOnly Property Attributes As ReadOnlyTypeAttributeCollection
    Get
```

Property Value

Type: ReadOnlyTypeAttributeCollection [▶ 1776]
The attributes or NULL if no Attributes are defined.

Implements

ITcAdsSymbol4.Attributes [▶ 563]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.4 TcAdsSymbolInfo.BitSize Property

Gets the size of this TcAdsSymbolInfo [▶ 780] in bits.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int BitSize { get; }
```

VB

```
Public ReadOnly Property BitSize As Integer
    Get
```

Property Value

Type: Int32
The size of the bit.

Implements

ITcAdsSymbol4.BitSize [▶ 564]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.5 TcAdsSymbolInfo.ByteSize Property

Gets the (aligned) size of of the Type/Instance in Bytes

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ByteSize { get; }
```

VB

```
Public ReadOnly Property ByteSize As Integer
    Get
```

Property Value

Type: Int32
The size of the byte.

Implements

ITcAdsSymbol4.ByteSize [▶ 564]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.6 TcAdsSymbolInfo.Category Property

Gets the Data Type Category

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeCategory Category { get; }
```

VB

```
Public ReadOnly Property Category As DataTypeCategory
    Get
```

Property Value

Type: DataTypeCategory [▶ 1189]
The category.

Implements

ITcAdsSymbol4.Category [▶ 565]

Exceptions

Exception	Condition
NotImplementedException	

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.7 TcAdsSymbolInfo.Comment Property

Gets the comment behind the variable declaration.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Comment { get; }
```

VB

```
Public ReadOnly Property Comment As String
    Get
```

Property Value

Type: String
Comment behind the variable declaration.

Implements

ITcAdsSymbol.Comment [▶ 551]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.8 TcAdsSymbolInfo.ContextMask Property

Gets the ContextMask of the symbol, indicating the task the variable belongs to. If ContextMask is not zero use CyclicInContext [▶ 401] or OnChangeInContext [▶ 401] to add notifications.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ContextMask { get; }
```

VB

```
Public ReadOnly Property ContextMask As Integer
    Get
```

Property Value

Type: Int32
The context mask.

Implements

ITcAdsSymbol2.ContextMask [▶ 555]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.9 TcAdsSymbolInfo.DataType Property

Gets the DataType object

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ITcAdsDataType DataType { get; }
```

VB

```
Public ReadOnly Property DataType As ITcAdsDataType
    Get
```

Property Value

Type: ITcAdsDataType [▶ 524]
The dataType if available, or NULL.

Implements

ITcAdsSymbol5.DataType [▶ 569]

Remarks

Tries to resolve the DataType if not cached.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.10 TcAdsSymbolInfo.DataTypeId Property

Data type of the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsDatatypeId DataTypeId { get; }
```

VB

```
Public ReadOnly Property DataTypeId As AdsDatatypeId
    Get
```

Property Value

Type: AdsDatatypeId [▶ 299]
Data type of the symbol.

Implements

ITcAdsSymbol5.DataTypeId [▶ 569]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.11 TcAdsSymbolInfo.HasRpcMethods Property

Gets a value indicating whether this instance has RPC methods (Struct types only)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool HasRpcMethods { get; }
```

VB

```
Public ReadOnly Property HasRpcMethods As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has RPC methods; otherwise, false.

Implements

ITcAdsSymbol4.HasRpcMethods [▶ 565]

Remarks

The DataType (Structure) must be marked with the PlcAttribute 'TcRpcEnable' to enable RpcMethods, otherwise RpcMethods are not passed through to the ADS symbolic information.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.12 TcAdsSymbolInfo.IndexGroup Property

Gets the index group of the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public long IndexGroup { get; }
```

VB

```
Public ReadOnly Property IndexGroup As Long
    Get
```

Property Value

Type: Int64
Index group of the symbol.

Implements

ITcAdsSymbol.IndexGroup [▶ 551]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.13 TcAdsSymbolInfo.IndexOffset Property

Gets the index offset of the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public long IndexOffset { get; }
```

VB

```
Public ReadOnly Property IndexOffset As Long
    Get
```

Property Value

Type: Int64
Index offset of the symbol.

Implements

ITcAdsSymbol.IndexOffset [▶ 552]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.14 TcAdsSymbolInfo.IsBitType Property

Indicates if the BitValue flag is set for the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsBitType { get; }
```

VB

```
Public ReadOnly Property IsBitType As Boolean
    Get
```

Property Value

Type: Boolean
true if is BitValue, otherwise false.

Implements

ITcAdsSymbol2.IsBitType [▶ 555]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.15 TcAdsSymbolInfo.IsOversamplingArray Property

Gets a value indicating whether this instance is oversampling array.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsOversamplingArray { get; }
```

VB

```
Public ReadOnly Property IsOversamplingArray As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is oversampling array; otherwise, false.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.16 TcAdsSymbolInfo.IsPersistent Property

Indicates if the Persistent flag is set for the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsPersistent { get; }
```

VB

```
Public ReadOnly Property IsPersistent As Boolean
    Get
```

Property Value

Type: Boolean
true if persistent, otherwise false.

Implements

ITcAdsSymbol2.IsPersistent [▶ 556]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.17 TcAdsSymbolInfo.IsReadOnly Property

Indicates if the ReadOnly flag is set for the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if ReadOnly, otherwise false.

Implements

ITcAdsSymbol2.IsReadOnly [▶ 556]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.18 TcAdsSymbolInfo.IsStatic Property

Indicates, that this symbol is a static symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsStatic { get; }
```

VB

```
Public ReadOnly Property IsStatic As Boolean
    Get
```

Property Value

Type: Boolean
Static indicator.

Implements

ITcAdsSymbol5.IsStatic [▶ 570]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.19 TcAdsSymbolInfo.IsTcComInterfacePointer Property

Indicates if the TcComInterfacePointer flag is set for the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsTcComInterfacePointer { get; }
```

VB

```
Public ReadOnly Property IsTcComInterfacePointer As Boolean
    Get
```

Property Value

Type: Boolean
treu if is TcComInterfacePointer, otherwise false.

Implements

ITcAdsSymbol2.IsTcComInterfacePointer [▶ 557]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.20 TcAdsSymbolInfo.IsTypeGuid Property

Indicates if the TypeGuid flag is set for the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsTypeGuid { get; }
```

VB

```
Public ReadOnly Property IsTypeGuid As Boolean
    Get
```

Property Value

Type: Boolean
true if is TypeGuid, otherwise false.

Implements

ITcAdsSymbol2.IsTypeGuid [▶ 557]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.21 TcAdsSymbolInfo.Name Property

Gets the name of the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Name { get; }
```

VB

```
Public ReadOnly Property Name As String
    Get
```

Property Value

Type: String
Name of the symbol.

Implements

ITcAdsSymbol.Name [▶ 552]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.22 TcAdsSymbolInfo.Parent Property

Gets the parent of this symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TcAdsSymbolInfo Parent { get; }
```

VB

```
Public ReadOnly Property Parent As TcAdsSymbolInfo
    Get
```

Property Value

Type: TcAdsSymbolInfo [▶ 780]
Parent of this symbol

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.23 TcAdsSymbolInfo.RpcMethods Property

Gets the RPC method descriptions

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyRpcMethodCollection RpcMethods { get; }
```

VB

```
Public ReadOnly Property RpcMethods As ReadOnlyRpcMethodCollection
    Get
```

Property Value

Type: ReadOnlyRpcMethodCollection [▶ 1762]
The RPC methods.

Implements

ITcAdsSymbol4.RpcMethods [▶ 566]

Remarks

The DataType (Structure) must be marked with the PlcAttribute 'TcRpcEnable' to enable RpcMethods, otherwise RpcMethods are not passed through to the ADS symbolic information.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.24 TcAdsSymbolInfo.ShortName Property

Gets the name of the symbol (short form without prefixed names of the parents).

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ShortName { get; }
```

VB

```
Public ReadOnly Property ShortName As String
    Get
```

Property Value

Type: String
Full name of the symbol.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.25 TcAdsSymbolInfo.Size Property

Gets the size of the symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Size { get; }
```

VB

```
Public ReadOnly Property Size As Integer
    Get
```

Property Value

Type: Int32
Size of the symbol.

Implements

ITcAdsSymbol.Size [▶ 553]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.26 TcAdsSymbolInfo.SubSymbols Property

Gets the sub symbols of this symbol as a collection of TcAdsSymbolInfo objects.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TcAdsSymbolInfoCollection SubSymbols { get; }
```

VB

```
Public ReadOnly Property SubSymbols As TcAdsSymbolInfoCollection
    Get
```

Property Value

Type: TcAdsSymbolInfoCollection [▶ 805]
The SubSymbol collection.

Implements

ITcAdsSymbolBrowser.SubSymbols [▶ 572]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.1.27 TcAdsSymbolInfo.TypeName Property

Gets the name of the symbol data type.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string TypeName { get; }
```

VB

```
Public ReadOnly Property TypeName As String
    Get
```

Property Value

Type: String
Name of the symbol data type.

Implements

ITcAdsSymbol5.TypeName [▶ 570]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.2 TcAdsSymbolInfo Methods

The TcAdsSymbolInfo [▶ 780] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 799]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 800]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IsRecursive [▶ 801]	Gets a value indicating whether this instance is recursive.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString [▶ 801]	Returns a String that represents this instance. (Overrides Object.ToString..)
	TryGetPointerRef [▶ 802]	Gets the Base data type of the pointer (the referenced type)
	TryGetReference [▶ 802]	Gets the referenced data type of the reference.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.2.1 TcAdsSymbolInfo.Equals Method

Equals

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool Equals(
    Object obj
)
```

VB

```
Public Overrides Function Equals (
    obj As Object
) As Boolean
```

Parameters

obj	Type: System.Object The object to compare with the current object.

Return Value

Type: Boolean
true if the specified Object is equal to this instance; otherwise, false.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.2.2 TcAdsSymbolInfo.GetHashCode Method

Gets the HashCode of the Address

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override int GetHashCode()
```

VB

```
Public Overrides Function GetHashCode As Integer
```

Return Value

Type: Int32
A hash code for this instance, suitable for use in hashing algorithms and data structures like a hash table.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.2.3 TcAdsSymbolInfo.IsRecursive Method

Gets a value indicating whether this instance is recursive.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsRecursive(
    IEnumerable<ITcAdsSymbol5> parents
)
```

VB

```
Public Function IsRecursive (
    parents As IEnumerable(Of ITcAdsSymbol5)
) As Boolean
```

Parameters

parents	Type: System.Collections.Generic.IEnumerable.ITcAdsSymbol5 [▶ 566]. The parents.

Field Value

Type: Boolean
true if this instance is recursive; otherwise, false.

Return Value

Type: Boolean
true if the specified parents is recursive; otherwise, false.

Implements

ITcAdsSymbol5.IsRecursive(IEnumerable.ITcAdsSymbol5.) [▶ 571]

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.2.4 TcAdsSymbolInfo.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.2.5 TcAdsSymbolInfo.TryGetPointerRef Method

Gets the Base data type of the pointer (the referenced type)

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetPointerRef(
    out string referencedType
)
```

VB

```
Public Function TryGetPointerRef (
    <OutAttribute> ByRef referencedType As String
) As Boolean
```

Parameters

referencedType	Type: System.String. Type of the reference.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.2.6 TcAdsSymbolInfo.TryGetReference Method

Gets the referenced data type of the reference.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetReference(
    out string referencedType
)
```

VB

```
Public Function TryGetReference (
    <OutAttribute> ByRef referencedType As String
) As Boolean
```

Parameters

referencedType	Type: System.String. Type of the referenced.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.3 TcAdsSymbolInfo Operators

The TcAdsSymbolInfo [▶ 780] type exposes the following members.

Operators

	Name	Description
	Equality [▶ 803]	Implements the == operator.
	Inequality [▶ 804]	Implements the != operator.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.3.1 TcAdsSymbolInfo.Equality Operator

Implements the == operator.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator ==(
    TcAdsSymbolInfo s1,
    TcAdsSymbolInfo s2
)
```

VB

```
Public Shared Operator = (
    s1 As TcAdsSymbolInfo,
    s2 As TcAdsSymbolInfo
) As Boolean
```

Parameters

s1	Type: TwinCAT.Ads.TcAdsSymbolInfo [▶ 780] Symbol 1.
s2	Type: TwinCAT.Ads.TcAdsSymbolInfo [▶ 780] Symbol 2.

Return Value

Type: Boolean
The result of the operator.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.62.3.2 TcAdsSymbolInfo.Inequality Operator

Implements the != operator.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator !=(
    TcAdsSymbolInfo s1,
    TcAdsSymbolInfo s2
)
```

VB

```
Public Shared Operator <> (
    s1 As TcAdsSymbolInfo,
    s2 As TcAdsSymbolInfo
) As Boolean
```

Parameters

s1	Type: TwinCAT.Ads.TcAdsSymbolInfo [▶ 780] The symbol 1.
s2	Type: TwinCAT.Ads.TcAdsSymbolInfo [▶ 780] The symbol 2.

Return Value

Type: Boolean
The result of the operator.

Reference

TcAdsSymbolInfo Class [▶ 780]

TwinCAT.Ads Namespace [▶ 108]

5.2.63 TcAdsSymbolInfoCollection Class

Represents a collection of TcAdsSymbolInfo objects.

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TcAdsSymbolInfoCollection
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class TcAdsSymbolInfoCollection : ICollection,
    IEnumerable
```

VB

```
Public Class TcAdsSymbolInfoCollection
    Implements ICollection, IEnumerable
```

The TcAdsSymbolInfoCollection type exposes the following members.

Properties

	Name	Description
	Count [▶ 806]	Gets the number of elements contained in the collection.
	IsSynchronized [▶ 807]	Gets a value indicating whether access to the collection is synchronized (thread-safe).
	Item [▶ 807]	Gets the element at the specified index.
	SyncRoot [▶ 808]	Gets an object that can be used to synchronize access to the collection.

Methods

	Name	Description
	CopyTo [▶ 809]	Copies the elements of the collection to an Array, starting at a particular Array index.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	GetEnumerator [▶ 810]	Returns an enumerator that can iterate through the symbols (TcAdsSymbolInfo [▶ 780]) in this collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetSymbol(Int32) [▶ 811]	Gets the Symbol by index
	GetSymbol(String) [▶ 811]	Gets the symbol by name.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.63.1 TcAdsSymbolInfoCollection Properties

The TcAdsSymbolInfoCollection [▶ 805] type exposes the following members.

Properties

	Name	Description
	Count [▶ 806]	Gets the number of elements contained in the collection.
	IsSynchronized [▶ 807]	Gets a value indicating whether access to the collection is synchronized (thread-safe).
	Item [▶ 807]	Gets the element at the specified index.
	SyncRoot [▶ 808]	Gets an object that can be used to synchronize access to the collection.

Reference

TcAdsSymbolInfoCollection Class [▶ 805]

TwinCAT.Ads Namespace [▶ 108]

5.2.63.1.1 TcAdsSymbolInfoCollection.Count Property

Gets the number of elements contained in the collection.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual int Count { get; }
```

VB

```
Public Overridable ReadOnly Property Count As Integer
    Get
```

Property Value

Type: Int32

Implements

ICollection.Count

Reference

TcAdsSymbolInfoCollection Class [▶ 805]

TwinCAT.Ads Namespace [▶ 108]

5.2.63.1.2 TcAdsSymbolInfoCollection.IsSynchronized Property

Gets a value indicating whether access to the collection is synchronized (thread-safe).

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual bool IsSynchronized { get; }
```

VB

```
Public Overridable ReadOnly Property IsSynchronized As Boolean
    Get
```

Property Value

Type: Boolean

Implements

ICollection.IsSynchronized

Reference

TcAdsSymbolInfoCollection Class [▶ 805]

TwinCAT.Ads Namespace [▶ 108]

5.2.63.1.3 TcAdsSymbolInfoCollection.Item Property

Gets the element at the specified index.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TcAdsSymbolInfo this[
    int index
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    index As Integer
) As TcAdsSymbolInfo
    Get
```

Parameters

index	Type: System.Int32

Property Value

Type: TcAdsSymbolInfo [▶ 780]

Reference

TcAdsSymbolInfoCollection Class [▶ 805]

TwinCAT.Ads Namespace [▶ 108]

5.2.63.1.4 TcAdsSymbolInfoCollection.SyncRoot Property

Gets an object that can be used to synchronize access to the collection.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual Object SyncRoot { get; }
```

VB

```
Public Overridable ReadOnly Property SyncRoot As Object
    Get
```

Property Value

Type: Object

Implements

ICollection.SyncRoot

Reference

TcAdsSymbolInfoCollection Class [▶ 805]

TwinCAT.Ads Namespace [▶ 108]

5.2.63.2 TcAdsSymbolInfoCollection Methods

The TcAdsSymbolInfoCollection [▶ 805] type exposes the following members.

Methods

	Name	Description
	CopyTo [▶ 809]	Copies the elements of the collection to an Array, starting at a particular Array index.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 810]	Returns an enumerator that can iterate through the symbols (TcAdsSymbolInfo [▶ 780]) in this collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetSymbol(Int32) [▶ 811]	Gets the Symbol by index
	GetSymbol(String) [▶ 811]	Gets the symbol by name.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TcAdsSymbolInfoCollection Class [▶ 805]

TwinCAT.Ads Namespace [▶ 108]

5.2.63.2.1 TcAdsSymbolInfoCollection.CopyTo Method

Copies the elements of the collection to an Array, starting at a particular Array index.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual void CopyTo(
    Array array,
    int index
)
```

VB

```
Public Overridable Sub CopyTo (
    array As Array,
    index As Integer
)
```

Parameters

array	Type: System.Array The one-dimensional Array that is the destination of the elements copied from the collection. The Array must have zero-based indexing.
index	Type: System.Int32 The zero-based index in array at which copying begins.

Implements

ICollection.CopyTo(Array, Int32)

Reference

TcAdsSymbolInfoCollection Class [▶ 805]

TwinCAT.Ads Namespace [▶ 108]

5.2.63.2.2 TcAdsSymbolInfoCollection.GetEnumerator Method

Returns an enumerator that can iterate through the symbols (TcAdsSymbolInfo [▶ 780]) in this collection.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual IEnumerator GetEnumerator()
```

VB

```
Public Overridable Function GetEnumerator As IEnumerator
```

Return Value

Type: IEnumerator
An IEnumerator that can be used to iterate through all symbols (TcAdsSymbolInfo [▶ 780]) in this collection.

Implements

IEnumerable.GetEnumerator.

Reference

TcAdsSymbolInfoCollection Class [▶ 805]

TwinCAT.Ads Namespace [▶ 108]

5.2.63.2.3 TcAdsSymbolInfoCollection.GetSymbol Method

Overload List

	Name	Description
	GetSymbol(Int32) [▶ 811]	Gets the Symbol by index
	GetSymbol(String) [▶ 811]	Gets the symbol by name.

Reference

TcAdsSymbolInfoCollection Class [▶ 805]

TwinCAT.Ads Namespace [▶ 108]

TcAdsSymbolInfoCollection.GetSymbol Method (Int32)

Gets the Symbol by index

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TcAdsSymbolInfo GetSymbol(
    int index
)
```

VB

```
Public Function GetSymbol (
    index As Integer
) As TcAdsSymbolInfo
```

Parameters

index	Type: System.Int32 The index.

Return Value

Type: TcAdsSymbolInfo [▶ 780]
TcAdsSymbolInfo if found, or NULL

Reference

TcAdsSymbolInfoCollection Class [▶ 805]

GetSymbol Overload [▶ 811]

TwinCAT.Ads Namespace [▶ 108]

TcAdsSymbolInfoCollection.GetSymbol Method (String)

Gets the symbol by name.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TcAdsSymbolInfo GetSymbol(
    string name
)
```

VB

```
Public Function GetSymbol (
    name As String
) As TcAdsSymbolInfo
```

Parameters

name	Type: System.String The name.

Return Value

Type: TcAdsSymbolInfo [▶ 780]
TcAdsSymbolInfo.

Remarks

Only located Symbols will be returned.

Reference

TcAdsSymbolInfoCollection Class [▶ 805]

GetSymbol Overload [▶ 811]

TwinCAT.Ads Namespace [▶ 108]

5.2.64 TcAdsSymbolInfoLoader Class

The class TcAdsSymbolInfoLoader is responsible for downloading the list of declared variables and the data types from an ADS Server.

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TcAdsSymbolInfoLoader
Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class TcAdsSymbolInfoLoader : IEnumerable
```

VB

```
Public Class TcAdsSymbolInfoLoader
    Implements IEnumerable
```

The TcAdsSymbolInfoLoader type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	FindSymbol [▶ 814]	Call this method to find a symbol in the list.
	GetDataTypes [▶ 815]	Gets the data types.
	GetEnumerator [▶ 815]	Returns an enumerator that can iterate through all symbols.
	GetFirstSymbol [▶ 816]	Call this method to get the first symbol.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetSymbolCount [▶ 817]	Call this method to get the number of declared symbols.
	GetSymbols [▶ 818]	Loads the declared symbols from the ADS device and returns them as a collection of TcAdsSymbolInfo objects.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Events

	Name	Description
	TypeResolveError [▶ 819]	Occurs when a typename cannot be resolved.
	TypesGenerated [▶ 819]	Occurs when new types are generated

Reference

TwinCAT.Ads Namespace [▶ 108]

5.2.64.1 TcAdsSymbolInfoLoader Methods

The TcAdsSymbolInfoLoader [▶ 812] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	FindSymbol [▶ 814]	Call this method to find a symbol in the list.
	GetDataTypes [▶ 815]	Gets the data types.
	GetEnumerator [▶ 815]	Returns an enumerator that can iterate through all symbols.
	GetFirstSymbol [▶ 816]	Call this method to get the first symbol.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetSymbolCount [▶ 817]	Call this method to get the number of declared symbols.
	GetSymbols [▶ 818]	Loads the declared symbols from the ADS device and returns them as a collection of TcAdsSymbolInfo objects.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TcAdsSymbolInfoLoader Class [▶ 812]

TwinCAT.Ads Namespace [▶ 108]

5.2.64.1.1 TcAdsSymbolInfoLoader.FindSymbol Method

Call this method to find a symbol in the list.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TcAdsSymbolInfo FindSymbol(
    string name
)
```

VB

```
Public Function FindSymbol (
    name As String
) As TcAdsSymbolInfo
```

Parameters

name	Type: System.String Name of the symbol	

Return Value

Type: TcAdsSymbolInfo [▶ 780]
TcAdsSymbolInfo.

Remarks

If no symbols have been loaded before, the symbols are loaded from the server, regardless of the parameter forceReload.

PLC

Please attend to the PLC Control that the 'Symbol-Download' under Project / Options / TwinCAT is activated. For further information please see the manual TwinCAT PLC Control.

NC

The Symbol download has to be activated at each axis. This can de done in the configuration dialog of the axis under ‚General'. The field ‚Create Symbols' has to be marked. See manual of the TwinCAT System Manager.

Reference

TcAdsSymbolInfoLoader Class [▶ 812]

TwinCAT.Ads Namespace [▶ 108]

5.2.64.1.2 TcAdsSymbolInfoLoader.GetDataTypes Method

Gets the data types.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyTcAdsDataTypeCollection GetDataTypes(
    bool forceReload
)
```

VB

```
Public Function GetDataTypes (
    forceReload As Boolean
) As ReadOnlyTcAdsDataTypeCollection
```

Parameters

forceReload	Type: System.Boolean if set to true [force reload].

Return Value

Type: ReadOnlyTcAdsDataTypeCollection [▶ 581]
ReadOnlyTcAdsDataTypeCollection.

Reference

TcAdsSymbolInfoLoader Class [▶ 812]

TwinCAT.Ads Namespace [▶ 108]

5.2.64.1.3 TcAdsSymbolInfoLoader.GetEnumerator Method

Returns an enumerator that can iterate through all symbols.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual IEnumerator GetEnumerator()
```

VB

```
Public Overridable Function GetEnumerator As IEnumerator
```

Return Value

Type: IEnumerator
An IEnumerator that can be used to iterate through all symbols.

Implements

IEnumerable.GetEnumerator.

Reference

TcAdsSymbolInfoLoader Class [▶ 812]

TwinCAT.Ads Namespace [▶ 108]

5.2.64.1.4 TcAdsSymbolInfoLoader.GetFirstSymbol Method

Call this method to get the first symbol.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TcAdsSymbolInfo GetFirstSymbol(
    bool forceReload
)
```

VB

```
Public Function GetFirstSymbol (
    forceReload As Boolean
) As TcAdsSymbolInfo
```

Parameters

forceReload	Type: System.Boolean True, if a (new) loading of the symbol information from the server is required.

Return Value

Type: TcAdsSymbolInfo [▶ 780]
Returns the first symbol or null if no symbols are available.

Remarks

If no symbols have been loaded before, the symbols are loaded from the server, regardless of the parameter forceReload.

PLC

Please attend to the PLC Control that the 'Symbol-Download' under Project / Options / TwinCAT is activated. For further information please see the manual TwinCAT PLC Control.

NC

The Symbol download has to be activated at each axis. This can de done in the configuration dialog of the axis under ‚General'. The field ‚Create Symbols' has to be marked. See manual of the TwinCAT System Manager.

Reference

TcAdsSymbolInfoLoader Class [▶ 812]

TwinCAT.Ads Namespace [▶ 108]

5.2.64.1.5 TcAdsSymbolInfoLoader.GetSymbolCount Method

Call this method to get the number of declared symbols.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int GetSymbolCount(
    bool forceReload
)
```

VB

```
Public Function GetSymbolCount (
    forceReload As Boolean
) As Integer
```

Parameters

forceReload	Type: System.Boolean True, if a (new) loading of the symbol information from the server is required.

Return Value

Type: Int32
Returns the number of symbols.

Remarks

If no symbols have been loaded before, the symbols are loaded from the server, regardless of the parameter forceReload.

PLC

Please attend to the PLC Control that the 'Symbol-Download' under Project / Options / TwinCAT is activated. For further information please see the manual TwinCAT PLC Control.

NC

The Symbol download has to be activated at each axis. This can de done in the configuration dialog of the axis under ‚General'. The field ‚Create Symbols' has to be marked. See manual of the TwinCAT System Manager.

Reference

TcAdsSymbolInfoLoader Class [▶ 812]

TwinCAT.Ads Namespace [▶ 108]

5.2.64.1.6 TcAdsSymbolInfoLoader.GetSymbols Method

Loads the declared symbols from the ADS device and returns them as a collection of TcAdsSymbolInfo objects.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TcAdsSymbolInfoCollection GetSymbols(
    bool forceReload
)
```

VB

```
Public Function GetSymbols (
    forceReload As Boolean
) As TcAdsSymbolInfoCollection
```

Parameters

forceReload	Type: System.Boolean True, if a (new) loading of the symbol information from the server is required.

Return Value

Type: TcAdsSymbolInfoCollection [▶ 805]
A collection of TcAdsSymbolInfo objects

Remarks

If no symbols have been loaded before, the symbols are loaded from the server, regardless of the parameter forceReload.

PLC

Please attend to the PLC Control that the 'Symbol-Download' under Project / Options / TwinCAT is activated. For further information please see the manual TwinCAT PLC Control.

NC

The Symbol download has to be activated at each axis. This can de done in the configuration dialog of the axis under ‚General'. The field ‚Create Symbols' has to be marked. See manual of the TwinCAT System Manager.

Reference

TcAdsSymbolInfoLoader Class [▶ 812]

TwinCAT.Ads Namespace [▶ 108]

5.2.64.2 TcAdsSymbolInfoLoader Events

The TcAdsSymbolInfoLoader [▶ 812] type exposes the following members.

Events

	Name	Description
	TypeResolveError [▶ 819]	Occurs when a typename cannot be resolved.
	TypesGenerated [▶ 819]	Occurs when new types are generated

Reference

TcAdsSymbolInfoLoader Class [▶ 812]

TwinCAT.Ads Namespace [▶ 108]

5.2.64.2.1 TcAdsSymbolInfoLoader.TypeResolveError Event

Occurs when a typename cannot be resolved.

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event EventHandler<DataTypeNameEventArgs> TypeResolveError
```

VB

```
Public Event TypeResolveError As EventHandler(Of DataTypeNameEventArgs)
```

Value

Type: System.EventHandler.DataTypeNameEventArgs [▶ 1203].

Reference

TcAdsSymbolInfoLoader Class [▶ 812]

TwinCAT.Ads Namespace [▶ 108]

5.2.64.2.2 TcAdsSymbolInfoLoader.TypesGenerated Event

Occurs when new types are generated

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event EventHandler<DataTypeEventArgs> TypesGenerated
```

VB

```
Public Event TypesGenerated As EventHandler(Of DataTypeEventArgs)
```

Value

Type: System.EventHandler.DataTypeEventArgs [▶ 1196].

Reference

TcAdsSymbolInfoLoader Class [▶ 812]

TwinCAT.Ads Namespace [▶ 108]

5.2.65 TransportProtocol Enumeration

Enum ADS TransportProtocol

Namespace: TwinCAT.Ads [▶ 108]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[FlagsAttribute]
public enum TransportProtocol
```

VB

```
<FlagsAttribute>
Public Enumeration TransportProtocol
```

Members

	Member name	Value	Description
	None	0	None / Uninitialized
	Router	1	ADS via Router
	TcpIp	2	ADS via TCP/IP (without router)
	All	3	Indicates that Router and TcpIp are appropriate (for establishing connections)

Reference

TwinCAT.Ads Namespace [▶ 108]

5.3 TwinCAT.Ads.Reactive Namespace

Reactive Extensions for the ADS Client. All types within are contained in the ADS companion package "Beckhoff.TwinCAT.Ads.Reactive" which must be referenced seperately. https://www.nuget.org/packages/Beckhoff.TwinCAT.Ads.Reactive/

Classes

	Class	Description
	AdsClientExtensions [▶ 821]	Extension class for TcAdsClient [▶ 625] respective IAdsConnection [▶ 469] to provide reactive ADS extensions.
	AnyTypeExtensions [▶ 835]	Extension class for TcAdsClient [▶ 625] respective IAdsConnection [▶ 469] to provide reactive ADS extensions (accessing symbol value sequences with the ANY_TYPE concept)
	Notification [▶ 862]	Provides data for AdsNotificationEvent of the class
	NotificationBase [▶ 865]	Base class for Notifications
	NotificationEx [▶ 872]	Provides data for AdsNotificationExEvent of the class
	SymbolNotification [▶ 874]	Provides data for AdsNotificationEvent of the class
	ValueSymbolExtensions [▶ 877]	Extension class for TcAdsClient [▶ 625] respective IAdsConnection [▶ 469] to provide reactive ADS extensions for accessing symbols that are loaded by the IAdsSymbolLoaderFactory

5.3.1 AdsClientExtensions Class

Extension class for TcAdsClient [▶ 625] respective IAdsConnection [▶ 469] to provide reactive ADS extensions.

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.Reactive.AdsClientExtensions
Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static class AdsClientExtensions
```

VB

```
<ExtensionAttribute>
Public NotInheritable Class AdsClientExtensions
```

The AdsClientExtensions type exposes the following members.

Methods

	Name	Description
	PollAdsState(IAdsConnection, IObservable.Unit.) [▶ 825]	Gets an observable sequence of AdsState [▶ 365]s via Polling.

	Name	Description
	PollAdsState(IAdsConnection, TimeSpan) [▶ 826]	Gets an observable sequence of AdsState [▶ 365]s via Polling.
	WhenAdsStateChanges [▶ 828]	Gets an observable sequence of AdsState [▶ 365]s.
	WhenNotification(IAdsConnection, ISymbol) [▶ 829]	Gets an observable sequence of Notification [▶ 862]s.
	WhenNotification(IAdsConnection, ISymbolCollection) [▶ 830]	Gets an observable sequence of Notification [▶ 862] objects.
	WhenNotification(IAdsConnection, ISymbol, NotificationSettings) [▶ 832]	Gets an observable sequence of Notification [▶ 862]s.
	WhenNotification(IAdsConnection, ISymbolCollection, NotificationSettings) [▶ 833]	Gets an observable sequence of Notification [▶ 862] objects.

Remarks

Reactive Extensions (Rx) are a library for composing asynchronous and event-based programs using observable sequences and LINQ-style query operators. Using Rx, developers represent asynchronous data streams with Observables, query asynchronous data streams using LINQ operators, and parameterize the concurrency in the asynchronous data streams using Schedulers. Simply put, Rx = Observables + LINQ + Schedulers. The ADS reactive extensions are build on top of this library to enable ADS Symbol and State Observables, seamlessly bound to the reactive extensions. To use the ADS reactive extensions the TwinCAT.Ads.Reactive Nuget package (or the included TwinCAT.Ads.Reactive.dll) must be referenced. (Beckhoff.TwinCAT.Ads.Reactive package on Nuget).

Examples

The following sample shows how observe Value changed Notifications with the reactive AdsClientExtensions

Observe changing ADS Symbols with reactive extensions.

```
// To Test the Observer run a project on the local PLC System (Port 851)

using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
```

```
    int eventCount = 1;

    // Reactive Notification Handler
    var valueObserver = Observer.Create<SymbolNotification>(not =>
    {
        Console.WriteLine(string.Format("{0} {1:u} {2} = '{3}' ({4})", eventCount+
+, not.TimeStamp, not.Symbol.InstancePath, not.Value, not.Symbol.DataType));
    }
    );

    // Collect the symbols that are registered as Notification sources for their changed values.

    SymbolCollection notificationSymbols = new SymbolCollection();
    IArrayInstance taskInfo = (IArrayInstance)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskI
nfo"];

    foreach(ISymbol element in taskInfo.Elements)
    {
    ISymbol cycleCount = element.SubSymbols["CycleCount"];
    ISymbol lastExecTime = element.SubSymbols["LastExecTime"];

    notificationSymbols.Add(cycleCount);
    notificationSymbols.Add(lastExecTime);
    }

    // Create a subscription for the first 200 Notifications on Symbol Value changes.
    IDisposable subscription = client.WhenNotification(notificationSymbols,NotificationSettings.Defa
ult).Take(200).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Examples

The following sample shows how observe AdsState [▶ 365] changed Notifications with the reactive AdsClientExtensions

Observe changing ADS states with reactive extensions.

```
// To Test the observer, Start/Stop the local PLC

using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.DefaultDynamic);

    // Reactive Notification Handler
    var valueObserver = Observer.Create<IList<AdsState>>(not =>
    {
        AdsState oldValue = not[0];
        AdsState newValue = not[1];

        Console.WriteLine(string.Format("Changed ADSState from '{0}' --
> '{1}!", oldValue, newValue));
    }
    );

    // Create a subscription for the AdsState change and buffering 2 Values (for oldValue --
> newValue output).
    IDisposable subscription = client.WhenAdsStateChanges().Buffer(2,1).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Reference

TwinCAT.Ads.Reactive Namespace [▶ 820]

TwinCAT.Ads.Reactive.AnyTypeExtensions [▶ 835]

TwinCAT.Ads.Reactive.ValueSymbolExtensions [▶ 877]

5.3.1.1 AdsClientExtensions Methods

The AdsClientExtensions [▶ 821] type exposes the following members.

Methods

	Name	Description
	PollAdsState(IAdsConnection, IObservable.Unit.) [▶ 825]	Gets an observable sequence of AdsState [▶ 365]s via Polling.
	PollAdsState(IAdsConnection, TimeSpan) [▶ 826]	Gets an observable sequence of AdsState [▶ 365]s via Polling.
	WhenAdsStateChanges [▶ 828]	Gets an observable sequence of AdsState [▶ 365]s.
	WhenNotification(IAdsConnection, ISymbol) [▶ 829]	Gets an observable sequence of Notification [▶ 862]s.
	WhenNotification(IAdsConnection, ISymbolCollection) [▶ 830]	Gets an observable sequence of Notification [▶ 862] objects.
	WhenNotification(IAdsConnection, ISymbol, NotificationSettings) [▶ 832]	Gets an observable sequence of Notification [▶ 862]s.
	WhenNotification(IAdsConnection, ISymbolCollection, NotificationSettings) [▶ 833]	Gets an observable sequence of Notification [▶ 862] objects.

Reference

AdsClientExtensions Class [▶ 821]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.1.1.1 AdsClientExtensions.PollAdsState Method

Overload List

	Name	Description
	PollAdsState(IAdsConnection, IObservable.Unit.) [▶ 825]	Gets an observable sequence of AdsState [▶ 365]s via Polling.
	PollAdsState(IAdsConnection, TimeSpan) [▶ 826]	Gets an observable sequence of AdsState [▶ 365]s via Polling.

Reference

AdsClientExtensions Class [▶ 821]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AdsClientExtensions.PollAdsState Method (IAdsConnection, IObservable.Unit.)

Gets an observable sequence of AdsState [▶ 365]s via Polling.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<AdsState> PollAdsState(
    this IAdsConnection client,
    IObservable<Unit> trigger
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollAdsState (
    client As IAdsConnection,
    trigger As IObservable(Of Unit)
) As IObservable(Of AdsState)
```

Parameters

client	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The client.
trigger	Type: System.IObservable.Unit. The polling trigger

Return Value

Type: IObservable.AdsState [▶ 365].
IObservable<AdsState>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Examples

The following sample shows how observe AdsState [▶ 365] via polling with the reactive AdsClientExtensions [▶ 821]

Observe changing ADS States with reactive extensions.

```
// To Test the observer, Start/Stop the local PLC

using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.DefaultDynamic);

    // Reactive Change Handler
    var valueObserver = Observer.Create<IList<AdsState>>(not =>
    {
    AdsState oldValue = not[0];
    AdsState newValue = not[1];

    Console.WriteLine(string.Format("Changed ADSState from '{0}' --> '{1}!", oldValue, newValue));
    }
    );

    // Create a subscription for the AdsState change and buffering 2 Values (for oldValue --
> newValue output).
    IDisposable subscription = client.PollAdsState(TimeSpan.FromMilliseconds(200)).Buffer(2, 1).Subs
cribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Reference

AdsClientExtensions Class [▶ 821]

PollAdsState Overload [▶ 825]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AdsClientExtensions.WhenAdsStateChanges(TcAdsClient) [▶ 828]

AdsClientExtensions.PollAdsState Method (IAdsConnection, TimeSpan)

Gets an observable sequence of AdsState [▶ 365]s via Polling.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<AdsState> PollAdsState(
    this IAdsConnection client,
    TimeSpan period
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollAdsState (
    client As IAdsConnection,
    period As TimeSpan
) As IObservable(Of AdsState)
```

Parameters

client	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The client.
period	Type: System.TimeSpan The period.

Return Value

Type: IObservable.AdsState [▶ 365].
IObservable<AdsState>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Examples

The following sample shows how observe AdsState [▶ 365] via polling with the reactive AdsClientExtensions [▶ 821]

Observe changing ADS States with reactive extensions.

```
// To Test the observer, Start/Stop the local PLC

using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.DefaultDynamic);

    // Reactive Change Handler
    var valueObserver = Observer.Create<IList<AdsState>>(not =>
    {
    AdsState oldValue = not[0];
    AdsState newValue = not[1];

    Console.WriteLine(string.Format("Changed ADSState from '{0}' --> '{1}!", oldValue, newValue));
    }
    );

    // Create a subscription for the AdsState change and buffering 2 Values (for oldValue --
> newValue output).
    IDisposable subscription = client.PollAdsState(TimeSpan.FromMilliseconds(200)).Buffer(2, 1).Subs
cribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Reference

AdsClientExtensions Class [▶ 821]

PollAdsState Overload [▶ 825]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AdsClientExtensions.WhenAdsStateChanges(TcAdsClient) [▶ 828]

5.3.1.1.2 AdsClientExtensions.WhenAdsStateChanges Method

Gets an observable sequence of AdsState [▶ 365]s.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<AdsState> WhenAdsStateChanges(
    this TcAdsClient client
)
```

VB

```
<ExtensionAttribute>
Public Shared Function WhenAdsStateChanges (
    client As TcAdsClient
) As IObservable(Of AdsState)
```

Parameters

client	Type: TwinCAT.Ads.TcAdsClient [▶ 625] The client.

Return Value

Type: IObservable.AdsState [▶ 365].
IObservable<AdsState>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type TcAdsClient [▶ 625]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Examples

The following sample shows how observe AdsState [▶ 365] changed Notifications with the reactive AdsClientExtensions [▶ 821]

Observe changing ADS States with reactive extensions.

```
// To Test the observer, Start/Stop the local PLC

using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.DefaultDynamic);

    // Reactive Notification Handler
    var valueObserver = Observer.Create<IList<AdsState>>(not =>
    {
        AdsState oldValue = not[0];
        AdsState newValue = not[1];
```

```
        Console.WriteLine(string.Format("Changed ADSState from '{0}' --
> '{1}!", oldValue, newValue));
    }
    );

    // Create a subscription for the AdsState change and buffering 2 Values (for oldValue --
> newValue output).
    IDisposable subscription = client.WhenAdsStateChanges().Buffer(2,1).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Reference

AdsClientExtensions Class [▶ 821]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AdsClientExtensions.PollAdsState(IAdsConnection, IObservable.Unit.) [▶ 825]

5.3.1.1.3 AdsClientExtensions.WhenNotification Method

Overload List

	Name	Description
	WhenNotification(IAdsConnection, ISymbol) [▶ 829]	Gets an observable sequence of Notification [▶ 862]s.
	WhenNotification(IAdsConnection, ISymbolCollection) [▶ 830]	Gets an observable sequence of Notification [▶ 862] objects.
	WhenNotification(IAdsConnection, ISymbol, NotificationSettings) [▶ 832]	Gets an observable sequence of Notification [▶ 862]s.
	WhenNotification(IAdsConnection, ISymbolCollection, NotificationSettings) [▶ 833]	Gets an observable sequence of Notification [▶ 862] objects.

Reference

AdsClientExtensions Class [▶ 821]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AdsClientExtensions.WhenNotification Method (IAdsConnection, ISymbol)

Gets an observable sequence of Notification [▶ 862]s.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<Notification> WhenNotification(
    this IAdsConnection client,
    ISymbol symbol
)
```

VB

```
<ExtensionAttribute>
Public Shared Function WhenNotification (
    client As IAdsConnection,
    symbol As ISymbol
) As IObservable(Of Notification)
```

Parameters

client	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The client.
symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The symbol.

Return Value

Type: IObservable.Notification [▶ 862].
IObservable<NotificationValue>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AdsClientExtensions Class [▶ 821]

WhenNotification Overload [▶ 829]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AdsClientExtensions.WhenNotification(IAdsConnection, ISymbolCollection, NotificationSettings) [▶ 833]

AdsClientExtensions.WhenNotification(IAdsConnection, ISymbol, NotificationSettings) [▶ 832]

AdsClientExtensions.WhenNotification Method (IAdsConnection, ISymbolCollection)

Gets an observable sequence of Notification [▶ 862] objects.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<SymbolNotification> WhenNotification(
    this IAdsConnection client,
    ISymbolCollection symbols
)
```

VB

```
<ExtensionAttribute>
Public Shared Function WhenNotification (
    client As IAdsConnection,
    symbols As ISymbolCollection
) As IObservable(Of SymbolNotification)
```

Parameters

client	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The client.
symbols	Type: TwinCAT.TypeSystem.ISymbolCollection [▶ 1640] The symbols.

Return Value

Type: IObservable.SymbolNotification [▶ 874].
IObservable<NotificationValue>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Examples

The following sample shows how observe Value changed Notifications with the reactive AdsClientExtensions [▶ 821]

Observe changing ADS Symbols with reactive extensions.

```
// To Test the Observer run a project on the local PLC System (Port 851)

using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);

    int eventCount = 1;

    // Reactive Notification Handler
    var valueObserver = Observer.Create<SymbolNotification>(not =>
    {
        Console.WriteLine(string.Format("{0} {1:u} {2} = '{3}' ({4})", eventCount+
+, not.TimeStamp, not.Symbol.InstancePath, not.Value, not.Symbol.DataType));
    }
    );

    // Collect the symbols that are registered as Notification sources for their changed values.

    SymbolCollection notificationSymbols = new SymbolCollection();
    IArrayInstance taskInfo = (IArrayInstance)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskI
nfo"];

    foreach(ISymbol element in taskInfo.Elements)
```

```
    {
    ISymbol cycleCount = element.SubSymbols["CycleCount"];
    ISymbol lastExecTime = element.SubSymbols["LastExecTime"];

    notificationSymbols.Add(cycleCount);
    notificationSymbols.Add(lastExecTime);
    }

    // Create a subscription for the first 200 Notifications on Symbol Value changes.
    IDisposable subscription = client.WhenNotification(notificationSymbols,NotificationSettings.Defa
ult).Take(200).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Reference

AdsClientExtensions Class [▶ 821]

WhenNotification Overload [▶ 829]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AdsClientExtensions.WhenNotification(IAdsConnection, ISymbol, NotificationSettings) [▶ 832]

AdsClientExtensions.WhenNotification Method (IAdsConnection, ISymbol, NotificationSettings)

Gets an observable sequence of Notification [▶ 862]s.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<SymbolNotification> WhenNotification(
    this IAdsConnection client,
    ISymbol symbol,
    NotificationSettings settings
)
```

VB

```
<ExtensionAttribute>
Public Shared Function WhenNotification (
    client As IAdsConnection,
    symbol As ISymbol,
    settings As NotificationSettings
) As IObservable(Of SymbolNotification)
```

Parameters

client	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The client.
symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The symbol to observe.
settings	Type: TwinCAT.Ads.NotificationSettings [▶ 573] Notification settings.

Return Value

Type: IObservable.SymbolNotification [▶ 874].
IObservable<NotificationValue>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Exceptions

Exception	Condition
ArgumentNullException	symbol
ArgumentOutOfRangeException	Symbol is not an IValueSymbol - symbol

Reference

AdsClientExtensions Class [▶ 821]

WhenNotification Overload [▶ 829]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AdsClientExtensions.WhenNotification(IAdsConnection, ISymbolCollection, NotificationSettings) [▶ 833]

AdsClientExtensions.WhenNotification Method (IAdsConnection, ISymbolCollection, NotificationSettings)

Gets an observable sequence of Notification [▶ 862] objects.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<SymbolNotification> WhenNotification(
    this IAdsConnection client,
    ISymbolCollection symbols,
    NotificationSettings settings
)
```

VB

```
<ExtensionAttribute>
Public Shared Function WhenNotification (
    client As IAdsConnection,
    symbols As ISymbolCollection,
    settings As NotificationSettings
) As IObservable(Of SymbolNotification)
```

Parameters

client	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The client.

symbols	Type: TwinCAT.TypeSystem.ISymbolCollection [▶ 1640] The symbols to observe.
settings	Type: TwinCAT.Ads.NotificationSettings [▶ 573] The Notification settings.

Return Value

Type: IObservable.SymbolNotification [▶ 874].
IObservable<NotificationValue>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Examples

The following sample shows how observe Value changed Notifications with the reactive AdsClientExtensions [▶ 821]

Observe changing ADS Symbols with reactive extensions.

```
// To Test the Observer run a project on the local PLC System (Port 851)

using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);

    int eventCount = 1;

    // Reactive Notification Handler
    var valueObserver = Observer.Create<SymbolNotification>(not =>
    {
        Console.WriteLine(string.Format("{0} {1:u} {2} = '{3}' ({4})", eventCount+
+, not.TimeStamp, not.Symbol.InstancePath, not.Value, not.Symbol.DataType));
    }
    );

    // Collect the symbols that are registered as Notification sources for their changed values.

    SymbolCollection notificationSymbols = new SymbolCollection();
    IArrayInstance taskInfo = (IArrayInstance)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskI
nfo"];

    foreach(ISymbol element in taskInfo.Elements)
    {
    ISymbol cycleCount = element.SubSymbols["CycleCount"];
    ISymbol lastExecTime = element.SubSymbols["LastExecTime"];

    notificationSymbols.Add(cycleCount);
    notificationSymbols.Add(lastExecTime);
    }

    // Create a subscription for the first 200 Notifications on Symbol Value changes.
    IDisposable subscription = client.WhenNotification(notificationSymbols,NotificationSettings.Defa
ult).Take(200).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Reference

AdsClientExtensions Class [▶ 821]

WhenNotification Overload [▶ 829]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.2 AnyTypeExtensions Class

Extension class for TcAdsClient [▶ 625] respective IAdsConnection [▶ 469] to provide reactive ADS extensions (accessing symbol value sequences with the ANY_TYPE concept)

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.Reactive.AnyTypeExtensions
Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static class AnyTypeExtensions
```

VB

```
<ExtensionAttribute>
Public NotInheritable Class AnyTypeExtensions
```

Methods

	Name	Description
	PollValues(IAdsConnection, String, Type, IObservable.Unit.) [▶ 849]	Polls the symbol values on timepoints where the polling observable streams data / triggers
	PollValues(IAdsConnection, String, Type, TimeSpan) [▶ 850]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues(IAdsConnection, String, Type, .Int32., TimeSpan) [▶ 853]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues(IAdsConnection, String, Type, IObservable.Unit., Func.Exception, Object.) [▶ 854]	Polls the symbol values on timepoints where the polling observable streams data / triggers
	PollValues(IAdsConnection, String,	Polls the symbol as value sequence of object values with a specified period time.

	Name	Description
	Type, TimeSpan, Func.Exception, Object.) [▶ 855]	
	PollValues(IAdsConnection, String, Type, .Int32., IObservable.Unit., Func.Exception, Object.) [▶ 856]	Polls the symbol values on time points where the polling observable streams data / triggers
	PollValues(IAdsConnection, String, Type, .Int32., TimeSpan, Func.Exception, Object.) [▶ 858]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues.T.(IAdsConnection, String, IObservable.Unit.) [▶ 842]	Polls the symbol values on timepoints where the polling observable streams data / triggers
	PollValues.T.(IAdsConnection, String, TimeSpan) [▶ 843]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues.T.(IAdsConnection, String, .Int32., IObservable.Unit.) [▶ 844]	Polls the symbol values on time points where the polling observable streams data / triggers
	PollValues.T.(IAdsConnection, String, .Int32., TimeSpan) [▶ 845]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues.T.(IAdsConnection, String, IObservable.Unit., Func.Exception, T.) [▶ 847]	Polls the symbol values on timepoints where the polling observable streams data / triggers
	PollValues.T.(IAdsConnection, String, TimeSpan, Func.Exception, T.) [▶ 848]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues.T.(IAdsConnection, String, .Int32.,	Polls the symbol values on timepoints where the polling observable streams data / triggers

	Name	Description
	IObservable.Unit., Func.Exception, T.) [▶ 851]	
	PollValues.T. (IAdsConnection, String, .Int32., TimeSpan, Func.Exception, T.) [▶ 852]	Polls the symbol as value sequence of object values with a specified period time.
	WriteValues.T. (IAdsConnection, String, IObservable.T.) [▶ 859]	Writes the sequence of values to the symbol specified by the instance path.
	WriteValues.T. (IAdsConnection, String, IObservable.T., Action.Exception.) [▶ 860]	Writes the sequence of values to the symbol specified by the instance path.

Remarks

Reactive Extensions (Rx) are a library for composing asynchronous and event-based programs using observable sequences and LINQ-style query operators. Using Rx, developers represent asynchronous data streams with Observables, query asynchronous data streams using LINQ operators, and parameterize the concurrency in the asynchronous data streams using Schedulers. Simply put, Rx = Observables + LINQ + Schedulers. The ADS reactive extensions are build on top of this library to enable ADS Symbol and State Observables, seamlessly bound to the reactive extensions. To use the ADS reactive extensions the TwinCAT.Ads.Reactive Nuget package (or the included TwinCAT.Ads.Reactive.dll) must be referenced. (Beckhoff.TwinCAT.Ads.Reactive package on Nuget).

Examples

Example1: Observe Value changed Notifications with the reactive AnyTypeExtensions

Observe a single changing ADS Symbols (Extended AdsNotifications, ANY_TYPE)

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Reactive Notification Handler
    var valueObserver = Observer.Create<ushort>(val =>
    {
    Console.WriteLine(string.Format("Value: {0}", val.ToString()));
    }
    );

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    IDisposable subscription = client.WhenNotification<ushort>("TwinCAT_SystemInfoVarList._TaskInfo.
CycleCount", NotificationSettings.Default).Take(20).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Examples

Example2: Polling ANY_TYPE values.

Observe changing ADS Symbols by polling (Read Polling) (ANY_TYPE)

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o.CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    // Take 20 Values in an Interval of 500ms
    IDisposable subscription = cycleCount.PollValues(TimeSpan.FromMilliseconds(500)).Take(20).Subscr
ibe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Examples

Write values sequentially.

Write sequences of values to the target (ANY_TYPE)

```
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information (Symbol 'i : INT' in PLC Global Variables list.
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol gvlIntSymbol = (IValueSymbol)symbolLoader.Symbols["GVL.i"];

    // Produces object (short) Values 0,1,2,3 ... in seconds period
    IObservable<object> timerObservable = Observable.Interval(TimeSpan.FromSeconds(1.0)).Select(i =>
 (object)(short)i);

    // Take 10 Values (0..9) and write them to GVL.i
    IDisposable dispose = gvlIntSymbol.WriteValues(timerObservable.Take(10));

    Console.ReadKey(); // Wait for Key press
    dispose.Dispose(); // Dispose the Subscription
}
```

Reference

TwinCAT.Ads.Reactive Namespace [▶ 820]

TwinCAT.Ads.Reactive.AdsClientExtensions [▶ 821]

TwinCAT.Ads.Reactive.ValueSymbolExtensions [▶ 877]

5.3.2.1 AnyTypeExtensions Methods

Methods

	Name	Description
	PollValues(IAdsConnection, String, Type, IObservable.Unit.) [▶ 849]	Polls the symbol values on timepoints where the polling observable streams data / triggers
	PollValues(IAdsConnection, String, Type, TimeSpan) [▶ 850]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues(IAdsConnection, String, Type, .Int32., TimeSpan) [▶ 853]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues(IAdsConnection, String, Type, IObservable.Unit., Func.Exception, Object.) [▶ 854]	Polls the symbol values on timepoints where the polling observable streams data / triggers
	PollValues(IAdsConnection, String, Type, TimeSpan, Func.Exception, Object.) [▶ 855]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues(IAdsConnection, String, Type, .Int32., IObservable.Unit., Func.Exception, Object.) [▶ 856]	Polls the symbol values on time points where the polling observable streams data / triggers
	PollValues(IAdsConnection, String, Type, .Int32., TimeSpan, Func.Exception, Object.) [▶ 858]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues.T.(IAdsConnection, String, IObservable.Unit.) [▶ 842]	Polls the symbol values on timepoints where the polling observable streams data / triggers

	Name	Description
	PollValues.T. (IAdsConnection, String, TimeSpan) [▶ 843]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues.T. (IAdsConnection, String, .Int32., IObservable.Unit.) [▶ 844]	Polls the symbol values on time points where the polling observable streams data / triggers
	PollValues.T. (IAdsConnection, String, .Int32., TimeSpan) [▶ 845]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues.T. (IAdsConnection, String, IObservable.Unit., Func.Exception, T.) [▶ 847]	Polls the symbol values on timepoints where the polling observable streams data / triggers
	PollValues.T. (IAdsConnection, String, TimeSpan, Func.Exception, T.) [▶ 848]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues.T. (IAdsConnection, String, .Int32., IObservable.Unit., Func.Exception, T.) [▶ 851]	Polls the symbol values on timepoints where the polling observable streams data / triggers
	PollValues.T. (IAdsConnection, String, .Int32., TimeSpan, Func.Exception, T.) [▶ 852]	Polls the symbol as value sequence of object values with a specified period time.
	WriteValues.T. (IAdsConnection, String, IObservable.T.) [▶ 859]	Writes the sequence of values to the symbol specified by the instance path.
	WriteValues.T. (IAdsConnection, String, IObservable.T., Action.Exception.) [▶ 860]	Writes the sequence of values to the symbol specified by the instance path.

Reference

AnyTypeExtensions Class [▶ 835]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.2.1.1 AnyTypeExtensions.PollValues Method

Overload List

	Name	Description
	PollValues.T. (IAdsConnection, String, IObservable.Unit.) [▶ 842]	Polls the symbol values on timepoints where the polling observable streams data / triggers
	PollValues.T. (IAdsConnection, String, TimeSpan) [▶ 843]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues.T. (IAdsConnection, String, .Int32., IObservable.Unit.) [▶ 844]	Polls the symbol values on time points where the polling observable streams data / triggers
	PollValues.T. (IAdsConnection, String, .Int32., TimeSpan) [▶ 845]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues.T. (IAdsConnection, String, IObservable.Unit., Func.Exception, T.) [▶ 847]	Polls the symbol values on timepoints where the polling observable streams data / triggers
	PollValues.T. (IAdsConnection, String, TimeSpan, Func.Exception, T.) [▶ 848]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues(IAdsConnection, String, Type, IObservable.Unit.) [▶ 849]	Polls the symbol values on timepoints where the polling observable streams data / triggers
	PollValues(IAdsConnection, String, Type, TimeSpan) [▶ 850]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues.T. (IAdsConnection, String, .Int32.,	Polls the symbol values on timepoints where the polling observable streams data / triggers

	Name	Description
	IObservable.Unit., Func.Exception, T.) [▶ 851]	
	PollValues.T. (IAdsConnection, String, .Int32., TimeSpan, Func.Exception, T.) [▶ 852]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues(IAdsConnection, String, Type, .Int32., TimeSpan) [▶ 853]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues(IAdsConnection, String, Type, IObservable.Unit., Func.Exception, Object.) [▶ 854]	Polls the symbol values on timepoints where the polling observable streams data / triggers
	PollValues(IAdsConnection, String, Type, TimeSpan, Func.Exception, Object.) [▶ 855]	Polls the symbol as value sequence of object values with a specified period time.
	PollValues(IAdsConnection, String, Type, .Int32., IObservable.Unit., Func.Exception, Object.) [▶ 856]	Polls the symbol values on time points where the polling observable streams data / triggers
	PollValues(IAdsConnection, String, Type, .Int32., TimeSpan, Func.Exception, Object.) [▶ 858]	Polls the symbol as value sequence of object values with a specified period time.

Reference

AnyTypeExtensions Class [▶ 835]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues.T. Method (IAdsConnection, String, IObservable.Unit.)

Polls the symbol values on timepoints where the polling observable streams data / triggers

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<T> PollValues<T>(
    this IAdsConnection connection,
    string instancePath,
    IObservable<Unit> trigger
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues(Of T) (
    connection As IAdsConnection,
    instancePath As String,
    trigger As IObservable(Of Unit)
) As IObservable(Of T)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
trigger	Type: System.IObservable.Unit. The Polling trigger

Type Parameters

T	The ANY_TYPE compatible .NET Type.

Return Value

Type: IObservable.T.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues.T. Method (IAdsConnection, String, TimeSpan)

Polls the symbol as value sequence of object values with a specified period time.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<T> PollValues<T>(
    this IAdsConnection connection,
    string instancePath,
    TimeSpan period
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues(Of T) (
    connection As IAdsConnection,
    instancePath As String,
    period As TimeSpan
) As IObservable(Of T)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
period	Type: System.TimeSpan The period.

Type Parameters

T	The ANY_TYPE compatible .NET Type.

Return Value

Type: IObservable.T.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues.T. Method (IAdsConnection, String, .Int32., IObservable.Unit.)

Polls the symbol values on time points where the polling observable streams data / triggers

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<T> PollValues<T>(
    this IAdsConnection connection,
    string instancePath,
    int[] args,
    IObservable<Unit> trigger
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues(Of T) (
    connection As IAdsConnection,
    instancePath As String,
    args As Integer(),
    trigger As IObservable(Of Unit)
) As IObservable(Of T)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
args	Type: .System.Int32. ANY_TYPE arguments
trigger	Type: System.IObservable.Unit. The Polling trigger

Type Parameters

T	The ANY_TYPE compatible .NET Type.

Return Value

Type: IObservable.T.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues.T. Method (IAdsConnection, String, .Int32., TimeSpan)

Polls the symbol as value sequence of object values with a specified period time.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<T> PollValues<T>(
    this IAdsConnection connection,
    string instancePath,
    int[] args,
    TimeSpan period
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues(Of T) (
    connection As IAdsConnection,
    instancePath As String,
    args As Integer(),
    period As TimeSpan
) As IObservable(Of T)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
args	Type: .System.Int32. ANY_TYPE arguments.
period	Type: System.TimeSpan The period.

Type Parameters

T	The ANY_TYPE compatible .NET Type.

Return Value

Type: IObservable.T.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Examples

Polling ANY_TYPE values.

Observe changing ADS Symbols by polling (Read Polling) (ANY_TYPE)

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o.CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
```

```
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    // Take 20 Values in an Interval of 500ms
    IDisposable subscription = cycleCount.PollValues(TimeSpan.FromMilliseconds(500)).Take(20).Subscr
ibe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues.T. Method (IAdsConnection, String, IObservable.Unit., Func.Exception, T.)

Polls the symbol values on timepoints where the polling observable streams data / triggers

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<T> PollValues<T>(
    this IAdsConnection connection,
    string instancePath,
    IObservable<Unit> trigger,
    Func<Exception, T> errorHandler
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues(Of T) (
    connection As IAdsConnection,
    instancePath As String,
    trigger As IObservable(Of Unit),
    errorHandler As Func(Of Exception, T)
) As IObservable(Of T)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
trigger	Type: System.IObservable.Unit. The Polling trigger
errorHandler	Type: System.Func.Exception, T. The error handler.

Type Parameters

T	The ANY_TYPE compatible .NET Type.

Return Value

Type: IObservable.T.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues.T. Method (IAdsConnection, String, TimeSpan, Func.Exception, T.)

Polls the symbol as value sequence of object values with a specified period time.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<T> PollValues<T>(
    this IAdsConnection connection,
    string instancePath,
    TimeSpan period,
    Func<Exception, T> errorHandler
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues(Of T) (
    connection As IAdsConnection,
    instancePath As String,
    period As TimeSpan,
    errorHandler As Func(Of Exception, T)
) As IObservable(Of T)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
period	Type: System.TimeSpan The period.
errorHandler	Type: System.Func.Exception, T. The error handler.

Type Parameters

T	The ANY_TYPE compatible .NET Type.

Return Value

Type: IObservable.T.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues Method (IAdsConnection, String, Type, IObservable.Unit.)

Polls the symbol values on timepoints where the polling observable streams data / triggers

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<Object> PollValues(
    this IAdsConnection connection,
    string instancePath,
    Type type,
    IObservable<Unit> trigger
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues (
    connection As IAdsConnection,
    instancePath As String,
    type As Type,
    trigger As IObservable(Of Unit)
) As IObservable(Of Object)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
type	Type: System.Type The data type of the symbol (ANY_TYPE)
trigger	Type: System.IObservable.Unit. The Polling trigger

Return Value

Type: IObservable.Object.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues Method (IAdsConnection, String, Type, TimeSpan)

Polls the symbol as value sequence of object values with a specified period time.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<Object> PollValues(
    this IAdsConnection connection,
    string instancePath,
    Type type,
    TimeSpan period
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues (
    connection As IAdsConnection,
    instancePath As String,
    type As Type,
    period As TimeSpan
) As IObservable(Of Object)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
type	Type: System.Type The data type of the symbol (ANY_TYPE)
period	Type: System.TimeSpan The period.

Return Value

Type: IObservable.Object.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues.T. Method (IAdsConnection, String, .Int32., IObservable.Unit., Func.Exception, T.)

Polls the symbol values on timepoints where the polling observable streams data / triggers

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<T> PollValues<T>(
    this IAdsConnection connection,
    string instancePath,
    int[] args,
    IObservable<Unit> trigger,
    Func<Exception, T> errorHandler
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues(Of T) (
    connection As IAdsConnection,
    instancePath As String,
    args As Integer(),
    trigger As IObservable(Of Unit),
    errorHandler As Func(Of Exception, T)
) As IObservable(Of T)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
args	Type: .System.Int32. ANY_TYPE arguments
trigger	Type: System.IObservable.Unit. The Polling trigger
errorHandler	Type: System.Func.Exception, T. The error handler.

Type Parameters

T	The ANY_TYPE compatible .NET Type.

Return Value

Type: IObservable.T.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues.T. Method (IAdsConnection, String, .Int32., TimeSpan, Func.Exception, T.)

Polls the symbol as value sequence of object values with a specified period time.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<T> PollValues<T>(
    this IAdsConnection connection,
    string instancePath,
    int[] args,
    TimeSpan period,
    Func<Exception, T> errorHandler
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues(Of T) (
    connection As IAdsConnection,
    instancePath As String,
    args As Integer(),
    period As TimeSpan,
    errorHandler As Func(Of Exception, T)
) As IObservable(Of T)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
args	Type: .System.Int32. ANY_TYPE arguments.
period	Type: System.TimeSpan The period.
errorHandler	Type: System.Func.Exception, T. The error handler.

Type Parameters

T	The ANY_TYPE compatible .NET Type.

Return Value

Type: IObservable.T.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Examples

Polling ANY_TYPE values.

Observe changing ADS Symbols by polling (Read Polling) (ANY_TYPE)

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o.CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    // Take 20 Values in an Interval of 500ms
    IDisposable subscription = cycleCount.PollValues(TimeSpan.FromMilliseconds(500)).Take(20).Subscr
ibe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues Method (IAdsConnection, String, Type, .Int32., TimeSpan)

Polls the symbol as value sequence of object values with a specified period time.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<Object> PollValues(
    this IAdsConnection connection,
    string instancePath,
    Type type,
    int[] args,
    TimeSpan period
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues (
    connection As IAdsConnection,
    instancePath As String,
    type As Type,
    args As Integer(),
    period As TimeSpan
) As IObservable(Of Object)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
type	Type: System.Type The data type of the symbol (ANY_TYPE)
args	Type: .System.Int32. The ANY_TYPE arguments.
period	Type: System.TimeSpan The period.

Return Value

Type: IObservable.Object.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues Method (IAdsConnection, String, Type, IObservable.Unit., Func.Exception, Object.)

Polls the symbol values on timepoints where the polling observable streams data / triggers

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<Object> PollValues(
    this IAdsConnection connection,
    string instancePath,
    Type type,
    IObservable<Unit> trigger,
    Func<Exception, Object> errorHandler
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues (
    connection As IAdsConnection,
    instancePath As String,
    type As Type,
    trigger As IObservable(Of Unit),
    errorHandler As Func(Of Exception, Object)
) As IObservable(Of Object)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
type	Type: System.Type The data type of the symbol (ANY_TYPE)
trigger	Type: System.IObservable.Unit. The Polling trigger
errorHandler	Type: System.Func.Exception, Object. The error handler.

Return Value

Type: IObservable.Object.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues Method (IAdsConnection, String, Type, TimeSpan, Func.Exception, Object.)

Polls the symbol as value sequence of object values with a specified period time.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<Object> PollValues(
    this IAdsConnection connection,
    string instancePath,
    Type type,
    TimeSpan period,
    Func<Exception, Object> errorHandler
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues (
    connection As IAdsConnection,
    instancePath As String,
    type As Type,
    period As TimeSpan,
    errorHandler As Func(Of Exception, Object)
) As IObservable(Of Object)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
type	Type: System.Type The data type of the symbol (ANY_TYPE)
period	Type: System.TimeSpan The period.
errorHandler	Type: System.Func.Exception, Object. The error handler.

Return Value

Type: IObservable.Object.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues Method (IAdsConnection, String, Type, .Int32., IObservable.Unit., Func.Exception, Object.)

Polls the symbol values on time points where the polling observable streams data / triggers

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<Object> PollValues(
    this IAdsConnection connection,
    string instancePath,
    Type type,
    int[] args,
    IObservable<Unit> trigger,
    Func<Exception, Object> errorHandler
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues (
    connection As IAdsConnection,
    instancePath As String,
    type As Type,
    args As Integer(),
    trigger As IObservable(Of Unit),
    errorHandler As Func(Of Exception, Object)
) As IObservable(Of Object)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
type	Type: System.Type The data type of the symbol (ANY_TYPE)
args	Type: .System.Int32. The ANY_TYPE arguments.
trigger	Type: System.IObservable.Unit. The Polling trigger
errorHandler	Type: System.Func.Exception, Object. The error handler.

Return Value

Type: IObservable.Object.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.PollValues Method (IAdsConnection, String, Type, .Int32., TimeSpan, Func.Exception, Object.)

Polls the symbol as value sequence of object values with a specified period time.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<Object> PollValues(
    this IAdsConnection connection,
    string instancePath,
    Type type,
    int[] args,
    TimeSpan period,
    Func<Exception, Object> errorHandler
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValues (
    connection As IAdsConnection,
    instancePath As String,
    type As Type,
    args As Integer(),
    period As TimeSpan,
    errorHandler As Func(Of Exception, Object)
) As IObservable(Of Object)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
type	Type: System.Type The data type of the symbol (ANY_TYPE)
args	Type: .System.Int32. The ANY_TYPE arguments.
period	Type: System.TimeSpan The period.
errorHandler	Type: System.Func.Exception, Object. The error handler.

Return Value

Type: IObservable.Object.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

AnyTypeExtensions Class [▶ 835]

PollValues Overload [▶ 841]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.2.1.2 AnyTypeExtensions.WriteValues Method

Overload List

	Name	Description
	WriteValues.T. (IAdsConnection, String, IObservable.T.) [▶ 859]	Writes the sequence of values to the symbol specified by the instance path.
	WriteValues.T. (IAdsConnection, String, IObservable.T., Action.Exception.) [▶ 860]	Writes the sequence of values to the symbol specified by the instance path.

Reference

AnyTypeExtensions Class [▶ 835]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.WriteValues.T. Method (IAdsConnection, String, IObservable.T.)

Writes the sequence of values to the symbol specified by the instance path.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IDisposable WriteValues<T>(
    this IAdsConnection connection,
    string instancePath,
    IObservable<T> valueSequence
)
```

VB

```
<ExtensionAttribute>
Public Shared Function WriteValues(Of T) (
    connection As IAdsConnection,
    instancePath As String,
    valueSequence As IObservable(Of T)
) As IDisposable
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.

instancePath	Type: System.String The instance path.
valueSequence	Type: System.IObservable.T. Value sequence (Any type).

Type Parameters

T

Return Value

Type: IDisposable
IDisposable.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Examples

Write values sequentially.

Write sequences of values to the target (ANY_TYPE)

```
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information (Symbol 'i : INT' in PLC Global Variables list.
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol gvlIntSymbol = (IValueSymbol)symbolLoader.Symbols["GVL.i"];

    // Produces object (short) Values 0,1,2,3 ... in seconds period
    IObservable<object> timerObservable = Observable.Interval(TimeSpan.FromSeconds(1.0)).Select(i =>
 (object)(short)i);

    // Take 10 Values (0..9) and write them to GVL.i
    IDisposable dispose = gvlIntSymbol.WriteValues(timerObservable.Take(10));

    Console.ReadKey(); // Wait for Key press
    dispose.Dispose(); // Dispose the Subscription
}
```

Reference

AnyTypeExtensions Class [▶ 835]

WriteValues Overload [▶ 859]

TwinCAT.Ads.Reactive Namespace [▶ 820]

AnyTypeExtensions.WriteValues.T. Method (IAdsConnection, String, IObservable.T., Action.Exception.)

Writes the sequence of values to the symbol specified by the instance path.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IDisposable WriteValues<T>(
    this IAdsConnection connection,
    string instancePath,
    IObservable<T> valueSequence,
    Action<Exception> errorHandler
)
```

VB

```
<ExtensionAttribute>
Public Shared Function WriteValues(Of T) (
    connection As IAdsConnection,
    instancePath As String,
    valueSequence As IObservable(Of T),
    errorHandler As Action(Of Exception)
) As IDisposable
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePath	Type: System.String The instance path.
valueSequence	Type: System.IObservable.T. Value sequence (Any type).
errorHandler	Type: System.Action.Exception. The error handler.

Type Parameters

T

Return Value

Type: IDisposable
IDisposable.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Examples

Write values sequentially.

Write sequences of values to the target (ANY_TYPE)

```
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information (Symbol 'i : INT' in PLC Global Variables list.
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol gvlIntSymbol = (IValueSymbol)symbolLoader.Symbols["GVL.i"];

    // Produces object (short) Values 0,1,2,3 ... in seconds period
    IObservable<object> timerObservable = Observable.Interval(TimeSpan.FromSeconds(1.0)).Select(i =>
 (object)(short)i);
```

```
    // Take 10 Values (0..9) and write them to GVL.i
    IDisposable dispose = gvlIntSymbol.WriteValues(timerObservable.Take(10));

    Console.ReadKey(); // Wait for Key press
    dispose.Dispose(); // Dispose the Subscription
}
```

Reference

AnyTypeExtensions Class [▶ 835]

WriteValues Overload [▶ 859]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.3 Notification Class

Provides data for AdsNotificationEvent of the class

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.Reactive.NotificationBase [▶ 865]
    TwinCAT.Ads.Reactive.Notification
      TwinCAT.Ads.Reactive.SymbolNotification [▶ 874]

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public class Notification : NotificationBase
```

VB

```
Public Class Notification
    Inherits NotificationBase
```

The Notification type exposes the following members.

Properties

	Name	Description
	NotificationHandle [▶ 868]	Gets the handle of the connection. (Inherited from NotificationBase [▶ 865].)
	RawValue [▶ 864]	Streams that holds the notification data.
	TimeStamp [▶ 868]	Gets the timestamp of this Notification. [▶ 336] (Inherited from NotificationBase [▶ 865].)
	UserData [▶ 869]	Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data. (Inherited from NotificationBase [▶ 865].)
	Value [▶ 869]	Gets the value of the Notification [▶ 865]. (Inherited from NotificationBase [▶ 865].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Fields

	Name	Description
	_bytes [▶ 865]	Raw Data
	notificationHandle [▶ 870]	Notification Handle (Inherited from NotificationBase [▶ 865].)
	timeStamp [▶ 871]	Notification Time Stamp (Inherited from NotificationBase [▶ 865].)
	userData [▶ 871]	User Data (Inherited from NotificationBase [▶ 865].)
	val [▶ 872]	The unmarshalled value. (Inherited from NotificationBase [▶ 865].)

Reference

TwinCAT.Ads.Reactive Namespace [▶ 820]

TwinCAT.Ads.TcAdsClient [▶ 625]

5.3.3.1 Notification Properties

The Notification [▶ 862] type exposes the following members.

Properties

	Name	Description
	NotificationHandle [▶ 868]	Gets the handle of the connection. (Inherited from NotificationBase [▶ 865].)
	RawValue [▶ 864]	Streams that holds the notification data.
	TimeStamp [▶ 868]	Gets the timestamp of this Notification. [▶ 336] (Inherited from NotificationBase [▶ 865].)
	UserData [▶ 869]	Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data. (Inherited from NotificationBase [▶ 865].)
	Value [▶ 869]	Gets the value of the Notification [▶ 865]. (Inherited from NotificationBase [▶ 865].)

Reference

Notification Class [▶ 862]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.3.1.1 Notification.RawValue Property

Streams that holds the notification data.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public byte[] RawValue { get; }
```

VB

```
Public ReadOnly Property RawValue As Byte()
    Get
```

Property Value

Type: .Byte.

Reference

Notification Class [▶ 862]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.3.2 Notification Methods

The Notification [▶ 862] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

Notification Class [▶ 862]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.3.3 Notification Fields

The Notification [▶ 862] type exposes the following members.

Fields

	Name	Description
	_bytes [▶ 865]	Raw Data
	notificationHandle [▶ 870]	Notification Handle (Inherited from NotificationBase [▶ 865].)
	timeStamp [▶ 871]	Notification Time Stamp (Inherited from NotificationBase [▶ 865].)
	userData [▶ 871]	User Data (Inherited from NotificationBase [▶ 865].)
	val [▶ 872]	The unmarshalled value. (Inherited from NotificationBase [▶ 865].)

Reference

Notification Class [▶ 862]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.3.3.1 Notification._bytes Field

Raw Data

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
protected byte[] _bytes
```

VB

```
Protected _bytes As Byte()
```

Field Value

Type: .Byte.

Reference

Notification Class [▶ 862]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4 NotificationBase Class

Base class for Notifications

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.Reactive.NotificationBase
    TwinCAT.Ads.Reactive.Notification [▶ 862]
    TwinCAT.Ads.Reactive.NotificationEx [▶ 872]
Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public abstract class NotificationBase
```

VB

```
Public MustInherit Class NotificationBase
```

The NotificationBase type exposes the following members.

Constructors

	Name	Description
	NotificationBase [▶ 867]	

Properties

	Name	Description
	NotificationHandle [▶ 868]	Gets the handle of the connection.
	TimeStamp [▶ 868]	Gets the timestamp of this Notification. [▶ 336]
	UserData [▶ 869]	Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data.
	Value [▶ 869]	Gets the value of the Notification.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Fields

	Name	Description
	notificationHandle [▶ 870]	Notification Handle
	timeStamp [▶ 871]	Notification Time Stamp
	userData [▶ 871]	User Data
	val [▶ 872]	The unmarshalled value.

Reference

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4.1 NotificationBase Constructor

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
protected NotificationBase()
```

VB

```
Protected Sub New
```

Reference

NotificationBase Class [▶ 865]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4.2 NotificationBase Properties

The NotificationBase [▶ 865] type exposes the following members.

Properties

	Name	Description
	NotificationHandle [▶ 868]	Gets the handle of the connection.
	TimeStamp [▶ 868]	Gets the timestamp of this Notification. [▶ 336]
	UserData [▶ 869]	Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data.
	Value [▶ 869]	Gets the value of the Notification [▶ 865].

Reference

NotificationBase Class [▶ 865]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4.2.1 NotificationBase.NotificationHandle Property

Gets the handle of the connection.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public uint NotificationHandle { get; }
```

VB

```
Public ReadOnly Property NotificationHandle As UInteger
    Get
```

Property Value

Type: UInt32

Reference

NotificationBase Class [▶ 865]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4.2.2 NotificationBase.TimeStamp Property

Gets the timestamp of this Notification. [▶ 336]

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public DateTimeOffset TimeStamp { get; }
```

VB

```
Public ReadOnly Property TimeStamp As DateTimeOffset
    Get
```

Property Value

Type: DateTimeOffset

Reference

NotificationBase Class [▶ 865]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4.2.3 NotificationBase.UserData Property

Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public Object UserData { get; }
```

VB

```
Public ReadOnly Property UserData As Object
    Get
```

Property Value

Type: Object

Reference

NotificationBase Class [▶ 865]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4.2.4 NotificationBase.Value Property

Gets the value of the Notification [▶ 865].

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public virtual Object Value { get; }
```

VB

```
Public Overridable ReadOnly Property Value As Object
    Get
```

Property Value

Type: Object
The value.

Reference

NotificationBase Class [▶ 865]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4.3 NotificationBase Methods

The NotificationBase [▶ 865] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

NotificationBase Class [▶ 865]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4.4 NotificationBase Fields

The NotificationBase [▶ 865] type exposes the following members.

Fields

	Name	Description
	notificationHandle [▶ 870]	Notification Handle
	timeStamp [▶ 871]	Notification Time Stamp
	userData [▶ 871]	User Data
	val [▶ 872]	The unmarshalled value.

Reference

NotificationBase Class [▶ 865]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4.4.1 NotificationBase.notificationHandle Field

Notification Handle

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
protected uint notificationHandle
```

VB

```
Protected notificationHandle As UInteger
```

Field Value

Type: UInt32

Reference

NotificationBase Class [▶ 865]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4.4.2 NotificationBase.timeStamp Field

Notification Time Stamp

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
protected DateTimeOffset timeStamp
```

VB

```
Protected timeStamp As DateTimeOffset
```

Field Value

Type: DateTimeOffset

Reference

NotificationBase Class [▶ 865]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4.4.3 NotificationBase.userData Field

User Data

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
protected Object userData
```

VB

```
Protected userData As Object
```

Field Value

Type: Object

Reference

NotificationBase Class [▶ 865]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.4.4.4 NotificationBase.val Field

The unmarshalled value.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
protected Object val
```

VB

```
Protected val As Object
```

Field Value

Type: Object

Reference

NotificationBase Class [▶ 865]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.5 NotificationEx Class

Provides data for AdsNotificationExEvent of the class

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.Reactive.NotificationBase [▶ 865]
    TwinCAT.Ads.Reactive.NotificationEx
Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public sealed class NotificationEx : NotificationBase
```

VB

```
Public NotInheritable Class NotificationEx
    Inherits NotificationBase
```

The NotificationEx type exposes the following members.

Properties

	Name	Description
	NotificationHandle [▶ 868]	Gets the handle of the connection. (Inherited from NotificationBase [▶ 865].)
	TimeStamp [▶ 868]	Gets the timestamp of this Notification. [▶ 336] (Inherited from NotificationBase [▶ 865].)
	UserData [▶ 869]	Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data. (Inherited from NotificationBase [▶ 865].)
	Value [▶ 869]	Gets the value of the Notification [▶ 865]. (Inherited from NotificationBase [▶ 865].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.Ads.Reactive Namespace [▶ 820]

TwinCAT.Ads.TcAdsClient [▶ 625]

5.3.5.1 NotificationEx Properties

The NotificationEx [▶ 872] type exposes the following members.

Properties

	Name	Description
	NotificationHandle [▶ 868]	Gets the handle of the connection. (Inherited from NotificationBase [▶ 865].)
	TimeStamp [▶ 868]	Gets the timestamp of this Notification. [▶ 336] (Inherited from NotificationBase [▶ 865].)
	UserData [▶ 869]	Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data. (Inherited from NotificationBase [▶ 865].)
	Value [▶ 869]	Gets the value of the Notification [▶ 865]. (Inherited from NotificationBase [▶ 865].)

Reference

NotificationEx Class [▶ 872]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.5.2 NotificationEx Methods

The NotificationEx [▶ 872] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

NotificationEx Class [▶ 872]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.6 SymbolNotification Class

Provides data for AdsNotificationEvent of the class

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.Reactive.NotificationBase [▶ 865]
    TwinCAT.Ads.Reactive.Notification [▶ 862]
      TwinCAT.Ads.Reactive.SymbolNotification
Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public sealed class SymbolNotification : Notification
```

VB

```
Public NotInheritable Class SymbolNotification
    Inherits Notification
```

The SymbolNotification type exposes the following members.

Properties

	Name	Description
	NotificationHandle [▶ 868]	Gets the handle of the connection. (Inherited from NotificationBase [▶ 865].)
	RawValue [▶ 864]	Streams that holds the notification data. (Inherited from Notification [▶ 862].)
	Symbol [▶ 875]	Gets the symbol of the SymbolNotification.
	TimeStamp [▶ 868]	Gets the timestamp of this Notification. [▶ 336] (Inherited from NotificationBase [▶ 865].)
	UserData [▶ 869]	Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data. (Inherited from NotificationBase [▶ 865].)

	Name	Description
	Value [▶ 876]	Gets the value of the Notification. (Overrides NotificationBase.Value [▶ 869].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.Ads.Reactive Namespace [▶ 820]

TwinCAT.Ads.TcAdsClient [▶ 625]

5.3.6.1 SymbolNotification Properties

The SymbolNotification [▶ 874] type exposes the following members.

Properties

	Name	Description
	NotificationHandle [▶ 868]	Gets the handle of the connection. (Inherited from NotificationBase [▶ 865].)
	RawValue [▶ 864]	Streams that holds the notification data. (Inherited from Notification [▶ 862].)
	Symbol [▶ 875]	Gets the symbol of the SymbolNotification [▶ 874].
	TimeStamp [▶ 868]	Gets the timestamp of this Notification. [▶ 336] (Inherited from NotificationBase [▶ 865].)
	UserData [▶ 869]	Gets the user object. This object is passed by to AddDeviceNotification and can be used to store data. (Inherited from NotificationBase [▶ 865].)
	Value [▶ 876]	Gets the value of the Notification [▶ 874]. (Overrides NotificationBase.Value [▶ 869].)

Reference

SymbolNotification Class [▶ 874]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.6.1.1 SymbolNotification.Symbol Property

Gets the symbol of the SymbolNotification [▶ 874].

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public ISymbol Symbol { get; }
```

VB

```
Public ReadOnly Property Symbol As ISymbol
    Get
```

Property Value

Type: ISymbol [▶ 1634]
The value symbol.

Reference

SymbolNotification Class [▶ 874]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.6.1.2 SymbolNotification.Value Property

Gets the value of the Notification [▶ 874].

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public override Object Value { get; }
```

VB

```
Public Overrides ReadOnly Property Value As Object
    Get
```

Property Value

Type: Object
The value.

Reference

SymbolNotification Class [▶ 874]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.6.2 SymbolNotification Methods

The SymbolNotification [▶ 874] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)

	Name	Description
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

SymbolNotification Class [▶ 874]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.7 ValueSymbolExtensions Class

Extension class for TcAdsClient [▶ 625] respective IAdsConnection [▶ 469] to provide reactive ADS extensions for accessing symbols that are loaded by the IAdsSymbolLoaderFactory

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.Reactive.ValueSymbolExtensions
Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static class ValueSymbolExtensions
```

VB

```
<ExtensionAttribute>
Public NotInheritable Class ValueSymbolExtensions
```

Methods

	Name	Description
	PollValuesAnnotated(IValueSymbol, IObservable.Unit.) [▶ 882]	Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence
	PollValuesAnnotated(IValueSymbol, TimeSpan) [▶ 883]	Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time.
	WhenValueChanged(IValueSymbol) [▶ 884]	Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed.

	Name	Description
	WhenValueChanged (IAdsConnection, IEnumerable.ISymbol.) [▶ 886]	Observable sequence of Values driven by ADS Notifications on the specified symbol.
	WriteValues(IValueSymbol, IObservable.Object.) [▶ 888]	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].
	WriteValues(IValueSymbol, IObservable.Object., Action.Exception.) [▶ 890]	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].
	WriteValues(IValueSymbol, IObservable.Object., CancellationToken) [▶ 890]	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].
	WriteValues(IValueSymbol, IObservable.Object., Action.Exception., CancellationToken) [▶ 891]	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].

Remarks

Reactive Extensions (Rx) are a library for composing asynchronous and event-based programs using observable sequences and LINQ-style query operators. Using Rx, developers represent asynchronous data streams with Observables, query asynchronous data streams using LINQ operators, and parameterize the concurrency in the asynchronous data streams using Schedulers. Simply put, Rx = Observables + LINQ + Schedulers. The ADS reactive extensions are build on top of this library to enable ADS Symbol and State Observables, seamlessly bound to the reactive extensions. To use the ADS reactive extensions the TwinCAT.Ads.Reactive Nuget package (or the included TwinCAT.Ads.Reactive.dll) must be referenced from All types within are contained in the ADS companion package "Beckhoff.TwinCAT.Ads.Reactive" which must be referenced separately. (Beckhoff.TwinCAT.Ads.Reactive package on Nuget).

Examples

The following sample shows how to observe Value changed Notifications with the reactive ValueSymbolExtensions from an IValueSymbol [▶ 1683].

Observe a single changing ADS Symbol (ADS Notifications)

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
```

```
o[1].CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    cycleCount.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 500, 5000); //
 optional: Change NotificationSettings on Symbol

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    IDisposable subscription = cycleCount.WhenValueChanged().Take(20).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Examples

The following sample shows how to observe Value changed Notifications with the reactive ValueSymbolExtensions from an DynamicSymbol [▶ 1329].

Observe a single changing ADS Symbol (ADS Notifications) with the dynamic language runtime (.NET DLR)

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o[1].CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    cycleCount.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 500, 5000); //
 optional: Change NotificationSettings on Symbol

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    IDisposable subscription = cycleCount.WhenValueChanged().Take(20).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Examples

The same for more than one IValueSymbol [▶ 1683].

Observe changing ADS Symbols (ADS Notifications)

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o.CycleCount"]; // UShort Type
```

```
    IValueSymbol lastExecTime = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskI
nfo.LastExecTime"]; // UInt Type

    SymbolCollection symbols = new SymbolCollection();
    symbols.Add(cycleCount);
    symbols.Add(lastExecTime);

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    cycleCount.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange,500,5000); // o
ptional: Change NotificationSettings on Symbol

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    IDisposable subscription = client.WhenValueChanged(symbols).Take(20).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Examples

Here, the values are polled in a specific time period and sequential Reads are triggered (in opposite to ADS Notification in the latter example)

Observe changing ADS Symbols by polling (Read Polling)

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o[1].CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    // Take 20 Values in an Interval of 500ms
    IDisposable subscription = cycleCount.PollValues(TimeSpan.FromMilliseconds(500)).Take(20).Subscr
ibe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Examples

In the following example it is demonstrated how to write Values sequentially to a IValueSymbol [▶ 1683] with the help of the reactive extensions.

Write sequences of values to the target

```
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information (Symbol 'i : INT' in PLC Global Variables list.
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["GVL.i"];
```

```
    // Produces object Values 0,1,2,3 ... in seconds period
    IObservable<object> timerObservable = Observable.Interval(TimeSpan.FromSeconds(1.0)).Select(i =>
 (object)(short)i);

    // Take 10 Values (0..9) and write them to GVL.i
    IDisposable dispose = cycleCount.WriteValues(timerObservable.Take(10));

    Console.ReadKey(); // Wait for Key press
    dispose.Dispose(); // Dispose the Subscription
}
```

Reference

TwinCAT.Ads.Reactive Namespace [▶ 820]

TwinCAT.Ads.Reactive.AdsClientExtensions [▶ 821]

TwinCAT.Ads.Reactive.AnyTypeExtensions [▶ 835]

5.3.7.1 ValueSymbolExtensions Methods

Methods

	Name	Description
	PollValuesAnnotated(IValueSymbol, IObservable.Unit.) [▶ 882]	Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence
	PollValuesAnnotated(IValueSymbol, TimeSpan) [▶ 883]	Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time.
	WhenValueChanged(IValueSymbol) [▶ 884]	Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed.
	WhenValueChanged(IAdsConnection, IEnumerable.ISymbol.) [▶ 886]	Observable sequence of Values driven by ADS Notifications on the specified symbol.
	WriteValues(IValueSymbol, IObservable.Object.) [▶ 888]	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].
	WriteValues(IValueSymbol, IObservable.Object., Action.Exception.) [▶ 890]	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].
	WriteValues(IValueSymbol,	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].

	Name	Description
	IObservable.Object., CancellationToken) [▶ 890]	
	WriteValues(IValueSymbol, IObservable.Object., Action.Exception., CancellationToken) [▶ 891]	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].

Reference

ValueSymbolExtensions Class [▶ 877]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.7.1.1 ValueSymbolExtensions.PollValuesAnnotated Method

Overload List

	Name	Description
	PollValuesAnnotated(IValueSymbol, IObservable.Unit.) [▶ 882]	Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence
	PollValuesAnnotated(IValueSymbol, TimeSpan) [▶ 883]	Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time.

Reference

ValueSymbolExtensions Class [▶ 877]

TwinCAT.Ads.Reactive Namespace [▶ 820]

ValueSymbolExtensions.PollValuesAnnotated Method (IValueSymbol, IObservable.Unit.)

Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<ValueChangedArgs> PollValuesAnnotated(
    this IValueSymbol symbol,
    IObservable<Unit> trigger
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValuesAnnotated (
    symbol As IValueSymbol,
    trigger As IObservable(Of Unit)
) As IObservable(Of ValueChangedArgs)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.IValueSymbol [▶ 1683] The symbol.
trigger	Type: System.IObservable.Unit. The polling Trigger.

Return Value

Type: IObservable.ValueChangedArgs [▶ 1847].
IObservable<ValueChangedArgs>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IValueSymbol [▶ 1683]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

ValueSymbolExtensions Class [▶ 877]

PollValuesAnnotated Overload [▶ 882]

TwinCAT.Ads.Reactive Namespace [▶ 820]

ValueSymbolExtensions.PollValuesAnnotated Method (IValueSymbol, TimeSpan)

Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<ValueChangedArgs> PollValuesAnnotated(
    this IValueSymbol symbol,
    TimeSpan period
)
```

VB

```
<ExtensionAttribute>
Public Shared Function PollValuesAnnotated (
    symbol As IValueSymbol,
    period As TimeSpan
) As IObservable(Of ValueChangedArgs)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.IValueSymbol [▶ 1683] The symbol.

period	Type: System.TimeSpan The polling period/interval.

Return Value

Type: IObservable.ValueChangedArgs [▶ 1847].
IObservable<ValueChangedArgs>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IValueSymbol [▶ 1683]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

ValueSymbolExtensions Class [▶ 877]

PollValuesAnnotated Overload [▶ 882]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.3.7.1.2 ValueSymbolExtensions.WhenValueChanged Method

Overload List

	Name	Description
	WhenValueChanged (IValueSymbol) [▶ 884]	Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed.
	WhenValueChanged (IAdsConnection, IEnumerable.ISymbol.) [▶ 886]	Observable sequence of Values driven by ADS Notifications on the specified symbol.

Reference

ValueSymbolExtensions Class [▶ 877]

TwinCAT.Ads.Reactive Namespace [▶ 820]

ValueSymbolExtensions.WhenValueChanged Method (IValueSymbol)

Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<Object> WhenValueChanged(
    this IValueSymbol symbol
)
```

VB

```
<ExtensionAttribute>
Public Shared Function WhenValueChanged (
    symbol As IValueSymbol
) As IObservable(Of Object)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.IValueSymbol [▶ 1683] The symbol.

Return Value

Type: IObservable.Object.
IObservable<System.Object>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IValueSymbol [▶ 1683]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Examples

The following sample shows how to observe Value changed Notifications with the reactive ValueSymbolExtensions [▶ 877] from an IValueSymbol [▶ 1683].

Observe a single changing ADS Symbols (ADS Notifications)

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o[1].CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    cycleCount.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 500, 5000); //
 optional: Change NotificationSettings on Symbol

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    IDisposable subscription = cycleCount.WhenValueChanged().Take(20).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Reference

ValueSymbolExtensions Class [▶ 877]

WhenValueChanged Overload [▶ 884]

TwinCAT.Ads.Reactive Namespace [▶ 820]

ValueSymbolExtensions.WhenValueChanged Method (IAdsConnection, IEnumerable.ISymbol.)

Observable sequence of Values driven by ADS Notifications on the specified symbol.

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IObservable<Object> WhenValueChanged(
    this IAdsConnection connection,
    IEnumerable<ISymbol> symbols
)
```

VB

```
<ExtensionAttribute>
Public Shared Function WhenValueChanged (
    connection As IAdsConnection,
    symbols As IEnumerable(Of ISymbol)
) As IObservable(Of Object)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The ADS connection / ADS Client
symbols	Type: System.Collections.Generic.IEnumerable.ISymbol [▶ 1634]. The symbols to observe.

Return Value

Type: IObservable.Object.
IObservable<ValueChangedArgs>.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IAdsConnection [▶ 469]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Examples

The same for more than one IValueSymbol [▶ 1683].

Observe changing ADS Symbols (ADS Notifications)

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
```

```
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o.CycleCount"]; // UShort Type
    IValueSymbol lastExecTime = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskI
nfo.LastExecTime"]; // UInt Type

    SymbolCollection symbols = new SymbolCollection();
    symbols.Add(cycleCount);
    symbols.Add(lastExecTime);

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    cycleCount.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange,500,5000); // o
ptional: Change NotificationSettings on Symbol

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    IDisposable subscription = client.WhenValueChanged(symbols).Take(20).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Examples

The following sample shows how to observe Value changed Notifications with the reactive ValueSymbolExtensions [▶ 877] from an DynamicSymbol [▶ 1329].

Observe a single changing ADS Symbol (ADS Notifications) with the dynamic language runtime (.NET DLR)

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o[1].CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    cycleCount.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 500, 5000); //
 optional: Change NotificationSettings on Symbol

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    IDisposable subscription = cycleCount.WhenValueChanged().Take(20).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Reference

ValueSymbolExtensions Class [▶ 877]

WhenValueChanged Overload [▶ 884]

TwinCAT.Ads.Reactive Namespace [▶ 820]

Observable

5.3.7.1.3 ValueSymbolExtensions.WriteValues Method

Overload List

	Name	Description
	WriteValues(IValueSymbol, IObservable.Object.) [▶ 888]	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].
	WriteValues(IValueSymbol, IObservable.Object., Action.Exception.) [▶ 890]	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].
	WriteValues(IValueSymbol, IObservable.Object., CancellationToken) [▶ 890]	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].
	WriteValues(IValueSymbol, IObservable.Object., Action.Exception., CancellationToken) [▶ 891]	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].

Reference

ValueSymbolExtensions Class [▶ 877]

TwinCAT.Ads.Reactive Namespace [▶ 820]

ValueSymbolExtensions.WriteValues Method (IValueSymbol, IObservable.Object.)

Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IDisposable WriteValues(
    this IValueSymbol symbol,
    IObservable<Object> valueObservable
)
```

VB

```
<ExtensionAttribute>
Public Shared Function WriteValues (
    symbol As IValueSymbol,
    valueObservable As IObservable(Of Object)
) As IDisposable
```

Parameters

symbol	Type: TwinCAT.TypeSystem.IValueSymbol [▶ 1683] The symbol.
valueObservable	Type: System.IObservable.Object. Observable of Values.

Return Value

Type: IDisposable
IDisposable.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IValueSymbol [▶ 1683]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Examples

In the following example it is demonstrated how to write Values sequentially to a IValueSymbol [▶ 1683] with the help of the reactive extensions.

Write sequences of values to the target

```
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information (Symbol 'i : INT' in PLC Global Variables list.
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["GVL.i"];

    // Produces object Values 0,1,2,3 ... in seconds period
    IObservable<object> timerObservable = Observable.Interval(TimeSpan.FromSeconds(1.0)).Select(i =>
 (object)(short)i);

    // Take 10 Values (0..9) and write them to GVL.i
    IDisposable dispose = cycleCount.WriteValues(timerObservable.Take(10));

    Console.ReadKey(); // Wait for Key press
    dispose.Dispose(); // Dispose the Subscription
}
```

Reference

ValueSymbolExtensions Class [▶ 877]

WriteValues Overload [▶ 888]

TwinCAT.Ads.Reactive Namespace [▶ 820]

ValueSymbolExtensions.WriteValues Method (IValueSymbol, IObservable.Object., Action.Exception.)

Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static IDisposable WriteValues(
    this IValueSymbol symbol,
    IObservable<Object> valueObservable,
    Action<Exception> errorHandler
)
```

VB

```
<ExtensionAttribute>
Public Shared Function WriteValues (
    symbol As IValueSymbol,
    valueObservable As IObservable(Of Object),
    errorHandler As Action(Of Exception)
) As IDisposable
```

Parameters

symbol	Type: TwinCAT.TypeSystem.IValueSymbol [▶ 1683] The symbol.
valueObservable	Type: System.IObservable.Object. Observable of Values.
errorHandler	Type: System.Action.Exception. The error handler or NULL.

Return Value

Type: IDisposable
IDisposable.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IValueSymbol [▶ 1683]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

ValueSymbolExtensions Class [▶ 877]

WriteValues Overload [▶ 888]

TwinCAT.Ads.Reactive Namespace [▶ 820]

ValueSymbolExtensions.WriteValues Method (IValueSymbol, IObservable.Object., CancellationToken)

Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static void WriteValues(
    this IValueSymbol symbol,
    IObservable<Object> valueObservable,
    CancellationToken cancel
)
```

VB

```
<ExtensionAttribute>
Public Shared Sub WriteValues (
    symbol As IValueSymbol,
    valueObservable As IObservable(Of Object),
    cancel As CancellationToken
)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.IValueSymbol [▶ 1683] The symbol.
valueObservable	Type: System.IObservable.Object. Observable of Values.
cancel	Type: System.Threading.CancellationToken The cancellation token.

Return Value

Type:
IDisposable.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IValueSymbol [▶ 1683]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

ValueSymbolExtensions Class [▶ 877]

WriteValues Overload [▶ 888]

TwinCAT.Ads.Reactive Namespace [▶ 820]

ValueSymbolExtensions.WriteValues Method (IValueSymbol, IObservable.Object., Action.Exception., CancellationToken)

Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683].

Namespace: TwinCAT.Ads.Reactive [▶ 820]
Assembly: TwinCAT.Ads.Reactive (in TwinCAT.Ads.Reactive.dll) Version: 4.3.0.0 (4.3.7.0)

Syntax

C#

```
public static void WriteValues(
    this IValueSymbol symbol,
    IObservable<Object> valueObservable,
    Action<Exception> errorHandler,
    CancellationToken cancel
)
```

VB

```
<ExtensionAttribute>
Public Shared Sub WriteValues (
    symbol As IValueSymbol,
    valueObservable As IObservable(Of Object),
    errorHandler As Action(Of Exception),
    cancel As CancellationToken
)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.IValueSymbol [▶ 1683] The symbol.
valueObservable	Type: System.IObservable.Object. Observable of Values.
errorHandler	Type: System.Action.Exception. The error handler.
cancel	Type: System.Threading.CancellationToken The cancellation token.

Return Value

Type:
IDisposable.

Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type IValueSymbol [▶ 1683]. When you use instance method syntax to call this method, omit the first parameter. For more information, see Extension Methods (Visual Basic) or Extension Methods (C# Programming Guide).

Reference

ValueSymbolExtensions Class [▶ 877]

WriteValues Overload [▶ 888]

TwinCAT.Ads.Reactive Namespace [▶ 820]

5.4 TwinCAT.Ads.SumCommand Namespace

ADS offers powerful and fast communication to exchange any kind of information. It's possible to read single variables or complete arrays and structures with each one single ADS-API call. ADS Sum-Commands offer to read/write with one single ADS call multiple variables which are not structured within a linear memory, effectivly reducing roundtrips.

Classes

	Class	Description
	SumCreateHandles [▶ 897]	SumCommand for getting variable handles by a set of InstancePaths
	SumHandleRead [▶ 902]	Read (primitive, Any) values by Handle SumCommand.
	SumHandleWrite [▶ 905]	Write any (primitive) values by Handle SumCommand.
	SumReleaseHandles [▶ 909]	Release Handles SumCommand.
	SumSymbolRead [▶ 913]	Symbolic ADS Sum read access
	SumSymbolWrite [▶ 918]	Class for ADS Sum symbolic Write Access.

Interfaces

	Interface	Description
	ISumCommand [▶ 893]	Interface for SumCommands (Combined commands)

5.4.1 ISumCommand Interface

Interface for SumCommands (Combined commands)

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISumCommand
```

VB

```
Public Interface ISumCommand
```

The ISumCommand type exposes the following members.

Properties

	Name	Description
	Executed [▶ 895]	Gets a value indicating whether this ISumCommand was already executed.

	Name	Description
	Failed [▶ 895]	Gets a value indicating whether this ISumCommand failed.
	Result [▶ 896]	Gets teh AdsErrorCode [▶ 305] of the main SumCommand ADS Request
	SubResults [▶ 896]	Gets the sub results of the single Sub Requests.
	Succeeded [▶ 897]	Gets a value indicating whether this ISumCommand is succeeded.

Remarks

ADS offers powerful and fast communication to exchange any kind of information. It's possible to read single variables or complete arrays and structures with each one single ADS-API call. The ADS Sum Command command offers to read with one singale ADS call multiple variables which are not structured within a linear memory. As a result the ADS caller application (like scada Systems etc.) can extremly speed up cyclic polling : Sample :

- Until now : Polling 4000 single variables which are not in a linear area (like array / structure / fixed PLC address) would cause 4000 single Ads-ReadReq with each 1-2 ms protocol time. As a result the scanning of these variables take 4000ms-8000ms.
- New Ads-Command allows to read multiple variables with one single ADS-ReadReq : 4000 single variables are handled with e.g. 8 single Ads-ReadReq (each call requesting 500 variables) with each 1-2 ms protocol time. As a result the scanning of these variables take just few 10ms.

REQUIREMENTS AND IMPORTANT LIMITATIONS

ADS is just a transport layer, but there could be important side effects. So read this requirements and take care on limitations:

- **Version of target ADS Device** - ADS itself is just the transport layer, but the requested ADS device has to support the ADS-Command.
- **Bytes length of requested data** - Requesting a large list of values from variables is fine, but the requested data of the Ads-response (the data-byte-length) have to pass the AMS Router (size by default a 2048kb) So the caller has to limit the requested variables based on calculation of requested data-byte-length.
- **Number of Sub-ADS calls : Highly recommended to max. 500 !** - If the PLC is processing one ADS request, it will completely work on this single ADS request BEFORE starting neyt PLC cycle. As a result one single ADS request with 200.000 sub-Ads-requests would cause that PLC would collect and copy 200.000 variables into one single ADS response, before starting next PLC. So this large number of ads-sub-commands will jitter the PLC execution ! **We highly recommend to not request more than 500 Ads-Sub commands**

Reference

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.1.1 ISumCommand Properties

The ISumCommand [▶ 893] type exposes the following members.

Properties

	Name	Description
	Executed [▶ 895]	Gets a value indicating whether this ISumCommand [▶ 893] was already executed.
	Failed [▶ 895]	Gets a value indicating whether this ISumCommand [▶ 893] failed.
	Result [▶ 896]	Gets teh AdsErrorCode [▶ 305] of the main SumCommand ADS Request

	Name	Description
	SubResults [▶ 896]	Gets the sub results of the single Sub Requests.
	Succeeded [▶ 897]	Gets a value indicating whether this ISumCommand [▶ 893] is succeeded.

Reference

ISumCommand Interface [▶ 893]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.1.1.1 ISumCommand.Executed Property

Gets a value indicating whether this ISumCommand [▶ 893] was already executed.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool Executed { get; }
```

VB

```
ReadOnly Property Executed As Boolean
    Get
```

Property Value

Type: Boolean
true if executed; otherwise, false.

Reference

ISumCommand Interface [▶ 893]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.1.1.2 ISumCommand.Failed Property

Gets a value indicating whether this ISumCommand [▶ 893] failed.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool Failed { get; }
```

VB

```
ReadOnly Property Failed As Boolean
    Get
```

Property Value

Type: Boolean
true if failed; otherwise, false.

Reference

ISumCommand Interface [▶ 893]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.1.1.3 ISumCommand.Result Property

Gets teh AdsErrorCode [▶ 305] of the main SumCommand ADS Request

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsErrorCode Result { get; }
```

VB

```
ReadOnly Property Result As AdsErrorCode
    Get
```

Property Value

Type: AdsErrorCode [▶ 305]
The result.

Reference

ISumCommand Interface [▶ 893]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.1.1.4 ISumCommand.SubResults Property

Gets the sub results of the single Sub Requests.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AdsErrorCode[] SubResults { get; }
```

VB

```
ReadOnly Property SubResults As AdsErrorCode()
    Get
```

Property Value

Type: .AdsErrorCode [▶ 305].
The sub results.

Reference

ISumCommand Interface [▶ 893]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.1.1.5 ISumCommand.Succeeded Property

Gets a value indicating whether this ISumCommand [▶ 893] is succeeded.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool Succeeded { get; }
```

VB

```
ReadOnly Property Succeeded As Boolean
    Get
```

Property Value

Type: Boolean
true if succeeded; otherwise, false.

Reference

ISumCommand Interface [▶ 893]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.2 SumCreateHandles Class

SumCommand for getting variable handles by a set of InstancePaths

Inheritance Hierarchy

System.Object
  SumCommandWrapper.SumReadWrite.
    TwinCAT.Ads.SumCommand.SumCreateHandles
Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SumCreateHandles : SumCommandWrapper<SumReadWrite>
```

VB

```
Public Class SumCreateHandles
    Inherits SumCommandWrapper(Of SumReadWrite)
```

The SumCreateHandles type exposes the following members.

Constructors

	Name	Description
	SumCreateHandles(IAdsConnection, IList.String.) [▶ 899]	Initializes a new instance of the SumCreateHandles class.
	SumCreateHandles(IAdsConnection, .String.) [▶ 900]	Initializes a new instance of the SumCreateHandles class.

Methods

	Name	Description
	CreateHandles [▶ 901]	Creates the ADS handles.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Examples

Usage of Sum commands with handles (CreateHandles, Read, Write, ReleaseHandles)

```
/// <summary>
/// Defines the entry point of the application.
/// </summary>
/// <param name="args">The arguments.</param>
static void Main(string[] args)
{
    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for start:");
    Console.ReadLine();

    // Parse the command-line arguments
    AmsAddress address = ArgParser.Parse(args);

    using (TcAdsClient client = new TcAdsClient())
    {
    // Connect the AdsClient to the device target.
    client.Connect(address);

    string[] instancePathList = {"GVL.bVar", "GVL.iCount", "TwinCAT_SystemInfoVarList._AppInfo.Proje
ctName"};
    SumCreateHandles createHandlesCommand = new SumCreateHandles(client,instancePathList);

    uint[] handles = createHandlesCommand.CreateHandles();
    Type[] valueTypes = new Type[] { typeof(bool), typeof(short), typeof(string)};
```

```
    SumHandleRead readCommand = new SumHandleRead(client,handles,valueTypes);

    object[] readValues = readCommand.Read();

    for (int i = 0; i < instancePathList.Length; i++)
    {
        Console.WriteLine("Symbol: {0} (Value: {1}, Type: {2})",instancePathList[i],readValues[i].To
String(),valueTypes[i].Name);
    }

    // Sum Command Write
    SumHandleWrite writeCommand = new SumHandleWrite(client,handles,valueTypes);
    object[] writeValues = new object[] {true, (short) 42, "MyNewProjectName"};

    writeCommand.Write(writeValues);

    SumReleaseHandles releaseCommand = new SumReleaseHandles(client,handles);
    releaseCommand.ReleaseHandles();
    }

    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for leave:");
    Console.ReadLine();
}
```

Reference

TwinCAT.Ads.SumCommand Namespace [▶ 892]

TwinCAT.Ads.SumCommand.ISumCommand [▶ 893]

TwinCAT.Ads.SumCommand.SumReleaseHandles [▶ 909]

TwinCAT.Ads.SumCommand.SumHandleRead [▶ 902]

TwinCAT.Ads.SumCommand.SumHandleWrite [▶ 905]

5.4.2.1 SumCreateHandles Constructor

Overload List

	Name	Description
	SumCreateHandles(IAdsConnection, IList.String.) [▶ 899]	Initializes a new instance of the SumCreateHandles [▶ 897] class.
	SumCreateHandles(IAdsConnection, .String.) [▶ 900]	Initializes a new instance of the SumCreateHandles [▶ 897] class.

Reference

SumCreateHandles Class [▶ 897]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.2.1.1 SumCreateHandles Constructor (IAdsConnection, IList.String.)

Initializes a new instance of the SumCreateHandles [▶ 897] class.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SumCreateHandles(
    IAdsConnection connection,
    IList<string> instancePaths
)
```

VB

```
Public Sub New (
    connection As IAdsConnection,
    instancePaths As IList(Of String)
)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePaths	Type: System.Collections.Generic.IList.String. The instance paths.

Reference

SumCreateHandles Class [▶ 897]

SumCreateHandles Overload [▶ 899]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.2.1.2 SumCreateHandles Constructor (IAdsConnection, .String.)

Initializes a new instance of the SumCreateHandles [▶ 897] class.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SumCreateHandles(
    IAdsConnection connection,
    string[] instancePaths
)
```

VB

```
Public Sub New (
    connection As IAdsConnection,
    instancePaths As String()
)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
instancePaths	Type: .System.String. The instance paths.

Reference

SumCreateHandles Class [▶ 897]

SumCreateHandles Overload [▶ 899]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.2.2 SumCreateHandles Methods

The SumCreateHandles [▶ 897] type exposes the following members.

Methods

	Name	Description
	CreateHandles [▶ 901]	Creates the ADS handles.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

SumCreateHandles Class [▶ 897]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.2.2.1 SumCreateHandles.CreateHandles Method

Creates the ADS handles.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public uint[] CreateHandles()
```

VB

```
Public Function CreateHandles As UInteger()
```

Return Value

Type: .UInt32.
System.UInt32[].

Exceptions

Exception	Condition
AdsSumCommandException [▶ 384]	SumGetHandlesCommand failed!

Reference

SumCreateHandles Class [▶ 897]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.3 SumHandleRead Class

Read (primitive, Any) values by Handle SumCommand.

Inheritance Hierarchy

System.Object
  SumCommand
    SumRead
      TwinCAT.Ads.SumCommand.SumHandleRead
Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SumHandleRead : SumRead
```

VB

```
Public Class SumHandleRead
    Inherits SumRead
```

The SumHandleRead type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Read [▶ 904]	Reads the values.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryRead [▶ 905]	Tries to read the values of the .

Remarks

This is an ADS Sum Command to access values by handle information. It is always used in combination with and . By design (and in contrast to the symbolic access in SumSymbolRead [▶ 913], SumSymbolWrite [▶ 918]) this access method can act only with ADS ANY Type (Primitive) values (disadvantage). The Advantage is, that no symbolic information must be loaded before accessing the values, see samples:

Examples

Usage of Sum commands with handles (CreateHandles, Read, Write, ReleaseHandles)

```
/// <summary>
/// Defines the entry point of the application.
/// </summary>
/// <param name="args">The arguments.</param>
static void Main(string[] args)
{
    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for start:");
    Console.ReadLine();

    // Parse the command-line arguments
    AmsAddress address = ArgParser.Parse(args);

    using (TcAdsClient client = new TcAdsClient())
    {
    // Connect the AdsClient to the device target.
    client.Connect(address);

    string[] instancePathList = {"GVL.bVar", "GVL.iCount", "TwinCAT_SystemInfoVarList._AppInfo.Proje
ctName"};
    SumCreateHandles createHandlesCommand = new SumCreateHandles(client,instancePathList);

    uint[] handles = createHandlesCommand.CreateHandles();
    Type[] valueTypes = new Type[] { typeof(bool), typeof(short), typeof(string)};

    SumHandleRead readCommand = new SumHandleRead(client,handles,valueTypes);

    object[] readValues = readCommand.Read();

    for (int i = 0; i < instancePathList.Length; i++)
    {
        Console.WriteLine("Symbol: {0} (Value: {1}, Type: {2})",instancePathList[i],readValues[i].To
String(),valueTypes[i].Name);
    }

    // Sum Command Write
    SumHandleWrite writeCommand = new SumHandleWrite(client,handles,valueTypes);
    object[] writeValues = new object[] {true, (short) 42, "MyNewProjectName"};

    writeCommand.Write(writeValues);

    SumReleaseHandles releaseCommand = new SumReleaseHandles(client,handles);
    releaseCommand.ReleaseHandles();
    }

    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for leave:");
    Console.ReadLine();
}
```

Reference

TwinCAT.Ads.SumCommand Namespace [▶ 892]

TwinCAT.Ads.SumCommand.SumCreateHandles [▶ 897]

TwinCAT.Ads.SumCommand.SumReleaseHandles [▶ 909]

TwinCAT.Ads.SumCommand.ISumCommand [▶ 893]

TwinCAT.Ads.SumCommand.SumReleaseHandles [▶ 909]

TwinCAT.Ads.SumCommand.SumReleaseHandles [▶ 909]

TwinCAT.Ads.SumCommand.SumHandleWrite [▶ 905]

5.4.3.1 SumHandleRead Methods

The SumHandleRead [▶ 902] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Read [▶ 904]	Reads the values.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryRead [▶ 905]	Tries to read the values of the .

Reference

SumHandleRead Class [▶ 902]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.3.1.1 SumHandleRead.Read Method

Reads the values.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object[] Read()
```

VB

```
Public Function Read As Object()
```

Return Value

Type: .Object.
System.Object[].

Exceptions

Exception	Condition
AdsSumCommandException [▶ 384]	SumAnyReadByHandleCommand failed!

Reference

SumHandleRead Class [▶ 902]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.3.1.2 SumHandleRead.TryRead Method

Tries to read the values of the .

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryRead(
    out Object[] values,
    out AdsErrorCode[] returnCodes
)
```

VB

```
Public Function TryRead (
    <OutAttribute> ByRef values As Object(),
    <OutAttribute> ByRef returnCodes As AdsErrorCode()
) As AdsErrorCode
```

Parameters

values	Type: .System.Object.. The values.
returnCodes	Type: .TwinCAT.Ads.AdsErrorCode [▶ 305].. The return codes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

SumHandleRead Class [▶ 902]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

TwinCAT.Ads.SumCommand.SumHandleRead [▶ 902]

5.4.4 SumHandleWrite Class

Write any (primitive) values by Handle SumCommand.

Inheritance Hierarchy

System.Object
  SumCommand
    SumWrite
      TwinCAT.Ads.SumCommand.SumHandleWrite

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SumHandleWrite : SumWrite
```

VB

```
Public Class SumHandleWrite
    Inherits SumWrite
```

The SumHandleWrite type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryWrite [▶ 908]	Tries to write the values.
	Write [▶ 909]	Writes the values to the Symbols.

Remarks

This is an ADS Sum Command to access values by handle information. It is always used in combination with and . By design (and in contrast to the symbolic access in SumSymbolRead [▶ 913], SumSymbolWrite [▶ 918]) this access method can act only with ADS ANY Type (Primitive) values (disadvantage). The Advantage is, that no symbolic information must be loaded before accessing the values, see samples:

Examples

Usage of Sum commands with handles (CreateHandles, Read, Write, ReleaseHandles)

```
/// <summary>
/// Defines the entry point of the application.
/// </summary>
/// <param name="args">The arguments.</param>
static void Main(string[] args)
{
    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for start:");
    Console.ReadLine();

    // Parse the command-line arguments
```

```
    AmsAddress address = ArgParser.Parse(args);

    using (TcAdsClient client = new TcAdsClient())
    {
    // Connect the AdsClient to the device target.
    client.Connect(address);

    string[] instancePathList = {"GVL.bVar", "GVL.iCount", "TwinCAT_SystemInfoVarList._AppInfo.Proje
ctName"};
    SumCreateHandles createHandlesCommand = new SumCreateHandles(client,instancePathList);

    uint[] handles = createHandlesCommand.CreateHandles();
    Type[] valueTypes = new Type[] { typeof(bool), typeof(short), typeof(string)};

    SumHandleRead readCommand = new SumHandleRead(client,handles,valueTypes);

    object[] readValues = readCommand.Read();

    for (int i = 0; i < instancePathList.Length; i++)
    {
        Console.WriteLine("Symbol: {0} (Value: {1}, Type: {2})",instancePathList[i],readValues[i].To
String(),valueTypes[i].Name);
    }

    // Sum Command Write
    SumHandleWrite writeCommand = new SumHandleWrite(client,handles,valueTypes);
    object[] writeValues = new object[] {true, (short) 42, "MyNewProjectName"};

    writeCommand.Write(writeValues);

    SumReleaseHandles releaseCommand = new SumReleaseHandles(client,handles);
    releaseCommand.ReleaseHandles();
    }

    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for leave:");
    Console.ReadLine();
}
```

Reference

TwinCAT.Ads.SumCommand Namespace [▶ 892]

TwinCAT.Ads.SumCommand.SumCreateHandles [▶ 897]

TwinCAT.Ads.SumCommand.SumReleaseHandles [▶ 909]

TwinCAT.Ads.SumCommand.ISumCommand [▶ 893]

TwinCAT.Ads.SumCommand.SumReleaseHandles [▶ 909]

TwinCAT.Ads.SumCommand.SumReleaseHandles [▶ 909]

TwinCAT.Ads.SumCommand.SumHandleRead [▶ 902]

5.4.4.1 SumHandleWrite Methods

The SumHandleWrite [▶ 905] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)

	Name	Description
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryWrite [▶ 908]	Tries to write the values.
	Write [▶ 909]	Writes the values to the Symbols.

Reference

SumHandleWrite Class [▶ 905]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.4.1.1 SumHandleWrite.TryWrite Method

Tries to write the values.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWrite(
    Object[] values,
    out AdsErrorCode[] returnCodes
)
```

VB

```
Public Function TryWrite (
    values As Object(),
    <OutAttribute> ByRef returnCodes As AdsErrorCode()
) As AdsErrorCode
```

Parameters

values	Type: .System.Object. The values (ANY/Primitive types only).
returnCodes	Type: .TwinCAT.Ads.AdsErrorCode [▶ 305].. The return codes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

SumHandleWrite Class [▶ 905]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.4.1.2 SumHandleWrite.Write Method

Writes the values to the Symbols.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    Object[] values
)
```

VB

```
Public Sub Write (
    values As Object()
)
```

Parameters

values	Type: .System.Object. The Values (Any primitive types only):

Exceptions

Exception	Condition
AdsSumCommandException [▶ 384]	SumAnyWriteByHandleCommand failed!

Reference

SumHandleWrite Class [▶ 905]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.5 SumReleaseHandles Class

Release Handles SumCommand.

Inheritance Hierarchy

System.Object
  SumCommandWrapper.SumWrite.
    TwinCAT.Ads.SumCommand.SumReleaseHandles
Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SumReleaseHandles : SumCommandWrapper<SumWrite>
```

VB

```
Public Class SumReleaseHandles
    Inherits SumCommandWrapper(Of SumWrite)
```

The SumReleaseHandles type exposes the following members.

Constructors

	Name	Description
	SumReleaseHandles [▶ 911]	Initializes a new instance of the SumReleaseHandles class.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ReleaseHandles [▶ 912]	Releases the handles.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryReleaseHandles [▶ 913]	Tries to Release the Handles

Remarks

Releases the specified ADS handles. Usually used in conjunction with the SumCreateHandles [▶ 897] and the SumHandleRead [▶ 902] / SumHandleWrite [▶ 905] commands.

Examples

Usage of Sum commands with handles (CreateHandles, Read, Write, ReleaseHandles)

```
/// <summary>
/// Defines the entry point of the application.
/// </summary>
/// <param name="args">The arguments.</param>
static void Main(string[] args)
{
    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for start:");
    Console.ReadLine();

    // Parse the command-line arguments
    AmsAddress address = ArgParser.Parse(args);

    using (TcAdsClient client = new TcAdsClient())
    {
    // Connect the AdsClient to the device target.
    client.Connect(address);

    string[] instancePathList = {"GVL.bVar", "GVL.iCount", "TwinCAT_SystemInfoVarList._AppInfo.Proje
ctName"};
    SumCreateHandles createHandlesCommand = new SumCreateHandles(client,instancePathList);

    uint[] handles = createHandlesCommand.CreateHandles();
    Type[] valueTypes = new Type[] { typeof(bool), typeof(short), typeof(string)};

    SumHandleRead readCommand = new SumHandleRead(client,handles,valueTypes);
```

```
    object[] readValues = readCommand.Read();

    for (int i = 0; i < instancePathList.Length; i++)
    {
        Console.WriteLine("Symbol: {0} (Value: {1}, Type: {2})",instancePathList[i],readValues[i].To
String(),valueTypes[i].Name);
    }

    // Sum Command Write
    SumHandleWrite writeCommand = new SumHandleWrite(client,handles,valueTypes);
    object[] writeValues = new object[] {true, (short) 42, "MyNewProjectName"};

    writeCommand.Write(writeValues);

    SumReleaseHandles releaseCommand = new SumReleaseHandles(client,handles);
    releaseCommand.ReleaseHandles();
    }

    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for leave:");
    Console.ReadLine();
}
```

Reference

TwinCAT.Ads.SumCommand Namespace [▶ 892]

TwinCAT.Ads.SumCommand.ISumCommand [▶ 893]

TwinCAT.Ads.SumCommand.SumCreateHandles [▶ 897]

TwinCAT.Ads.SumCommand.SumHandleRead [▶ 902]

TwinCAT.Ads.SumCommand.SumHandleWrite [▶ 905]

5.4.5.1 SumReleaseHandles Constructor

Initializes a new instance of the SumReleaseHandles [▶ 909] class.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SumReleaseHandles(
    IAdsConnection connection,
    uint[] serverHandles
)
```

VB

```
Public Sub New (
    connection As IAdsConnection,
    serverHandles As UInteger()
)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The connection.
serverHandles	Type: .System.UInt32. The handles.

Reference

SumReleaseHandles Class [▶ 909]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.5.2 SumReleaseHandles Methods

The SumReleaseHandles [▶ 909] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ReleaseHandles [▶ 912]	Releases the handles.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryReleaseHandles [▶ 913]	Tries to Release the Handles

Reference

SumReleaseHandles Class [▶ 909]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.5.2.1 SumReleaseHandles.ReleaseHandles Method

Releases the handles.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void ReleaseHandles()
```

VB

```
Public Sub ReleaseHandles
```

Exceptions

Exception	Condition
AdsSumCommandException [▶ 384]	SumReleaseHandlesCommand failed!

Reference

SumReleaseHandles Class [▶ 909]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.5.2.2 SumReleaseHandles.TryReleaseHandles Method

Tries to Release the Handles

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryReleaseHandles(
    out AdsErrorCode[] returnCodes
)
```

VB

```
Public Function TryReleaseHandles (
    <OutAttribute> ByRef returnCodes As AdsErrorCode()
) As AdsErrorCode
```

Parameters

returnCodes	Type: .TwinCAT.Ads.AdsErrorCode [▶ 305].. The return codes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Reference

SumReleaseHandles Class [▶ 909]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.6 SumSymbolRead Class

Symbolic ADS Sum read access

Inheritance Hierarchy

System.Object
  SumCommandWrapper.SumRead.
    SumSymbolCommand.SumRead.

TwinCAT.Ads.SumCommand.SumSymbolRead
Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SumSymbolRead : SumSymbolCommand<SumRead>
```

VB

```
Public Class SumSymbolRead
    Inherits SumSymbolCommand(Of SumRead)
```

The SumSymbolRead type exposes the following members.

Constructors

	Name	Description
	SumSymbolRead [▶ 916]	Initializes a new instance of the SumSymbolRead class.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Read [▶ 917]	Reads the Values.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryRead [▶ 918]	Tries to read the specified symbols.

Remarks

The SumSymbolRead implements symbolic read access with automatic (dynamic) value marshalling. The advantage of the symbolic access is (in contrast to the handle access classes SumHandleRead [▶ 902],SumHandleWrite [▶ 905]) that all type information is available when using this ADS Sum Command. The disadvantage is, that the Symbolic information must be loaded beforehand, see examples.

Examples

Usage of SumSymbolRead/SumSymbolWrite with AdsSession

```
/// <summary>
/// Defines the entry point of the application.
/// </summary>
/// <param name="args">The arguments.</param>
static void Main(string[] args)
{
    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for start:");
    Console.ReadLine();
```

```
    // Parse the command-line arguments
    AmsAddress address = ArgParser.Parse(args);

    using (AdsSession session = new AdsSession(address))
    {
    // Connect to the device target.
    AdsConnection connection = (AdsConnection)session.Connect();

    // Load symbolic information
    ReadOnlySymbolCollection symbols = session.SymbolServer.Symbols;

    ISymbol bVar1 = symbols["GVL.bVar1"];
    ISymbol bVar2 = symbols["GVL.bVar2"];
    ISymbol projectName = symbols["TwinCAT_SystemInfoVarList._AppInfo.ProjectName"];

    SymbolCollection coll = new SymbolCollection() {bVar1, bVar2, projectName};

    // Sum Command Read
    SumSymbolRead readCommand = new SumSymbolRead(connection,coll);
    object[] values = readCommand.Read();

    for (int i = 0; i < coll.Count; i++)
    {
        Console.WriteLine("Symbol: {0} (Value: {1}, Type: {2})",coll[i].InstancePath,values[i].ToStr
ing(),values[i].GetType().Name);
    }

    // Sum Command Write
    SumSymbolWrite writeCommand = new SumSymbolWrite(connection,coll);
    object[] writeValues = new object[] {true, (short) 42, "MyNewProjectName"};

    writeCommand.Write(writeValues);
    }

    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for leave:");
    Console.ReadLine();
}
```

Examples

Usage of SumSymbolRead/SumSymbolWrite with TcAdsClient

```
/// <summary>
/// Defines the entry point of the application.
/// </summary>
/// <param name="args">The arguments.</param>
static void Main(string[] args)
{
    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for start:");
    Console.ReadLine();

    // Parse the command-line arguments
    AmsAddress address = ArgParser.Parse(args);

    using (TcAdsClient client = new TcAdsClient())
    {
    // Connect the AdsClient to the device target.
    client.Connect(address);

    // Load symbolic information
    ISymbolLoader loader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    ReadOnlySymbolCollection allSymbols = loader.Symbols;

    ISymbol bVar1 = allSymbols["GVL.bVar1"];
    ISymbol bVar2 = allSymbols["GVL.iCount"];
    ISymbol projectName = allSymbols["TwinCAT_SystemInfoVarList._AppInfo.ProjectName"];

    SymbolCollection symbols = new SymbolCollection() {bVar1, bVar2, projectName};

    // Sum Command Read
    SumSymbolRead readCommand = new SumSymbolRead(client,symbols);
    object[] values = readCommand.Read();

    for (int i = 0; i < symbols.Count; i++)
    {
```

```
        Console.WriteLine("Symbol: {0} (Value: {1}, Type: {2})",symbols[i].InstancePath,values[i].To
String(),values[i].GetType().Name);
    }

    // Sum Command Write
    SumSymbolWrite writeCommand = new SumSymbolWrite(client,symbols);
    object[] writeValues = new object[] {true, (short) 42, "MyNewProjectName"};

    writeCommand.Write(writeValues);
    }

    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for leave:");
    Console.ReadLine();
}
```

Reference

TwinCAT.Ads.SumCommand Namespace [▶ 892]

TwinCAT.Ads.SumCommand.ISumCommand [▶ 893]

TwinCAT.Ads.SumCommand.SumSymbolWrite [▶ 918]

5.4.6.1 SumSymbolRead Constructor

Initializes a new instance of the SumSymbolRead [▶ 913] class.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SumSymbolRead(
    IAdsConnection connection,
    IList<ISymbol> symbols
)
```

VB

```
Public Sub New (
    connection As IAdsConnection,
    symbols As IList(Of ISymbol)
)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The TcAdsClient or ADS Connection object
symbols	Type: System.Collections.Generic.IList.ISymbol [▶ 1634]. The symbols to read

Reference

SumSymbolRead Class [▶ 913]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.6.2 SumSymbolRead Methods

The SumSymbolRead [▶ 913] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Read [▶ 917]	Reads the Values.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryRead [▶ 918]	Tries to read the specified symbols.

Reference

SumSymbolRead Class [▶ 913]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.6.2.1 SumSymbolRead.Read Method

Reads the Values.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object[] Read()
```

VB

```
Public Function Read As Object()
```

Return Value

Type: .Object.
System.Object[].

Exceptions

Exception	Condition
AdsSumCommandException [▶ 384]	SumSymbolRead failed!

Remarks

The return values are automatically marshalled to their appropriate .NET types.

Reference

SumSymbolRead Class [▶ 913]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.6.2.2 SumSymbolRead.TryRead Method

Tries to read the specified symbols.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryRead(
    out Object[] values,
    out AdsErrorCode[] returnCodes
)
```

VB

```
Public Function TryRead (
    <OutAttribute> ByRef values As Object(),
    <OutAttribute> ByRef returnCodes As AdsErrorCode()
) As AdsErrorCode
```

Parameters

values	Type: .System.Object.. The values.
returnCodes	Type: .TwinCAT.Ads.AdsErrorCode [▶ 305].. The return codes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Remarks

The returned values are automatically marshalled to their appropriate .NET types.

Reference

SumSymbolRead Class [▶ 913]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.7 SumSymbolWrite Class

Class for ADS Sum symbolic Write Access.

Inheritance Hierarchy

System.Object
  SumCommandWrapper.SumWrite.
    SumSymbolCommand.SumWrite.

TwinCAT.Ads.SumCommand.SumSymbolWrite
Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SumSymbolWrite : SumSymbolCommand<SumWrite>
```

VB

```
Public Class SumSymbolWrite
    Inherits SumSymbolCommand(Of SumWrite)
```

The SumSymbolWrite type exposes the following members.

Constructors

	Name	Description
	SumSymbolWrite [▶ 921]	Initializes a new instance of the SumSymbolWrite class.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryWrite [▶ 922]	Tries the write.
	Write [▶ 923]	Writes the specified values.

Remarks

The SumSymbolWrite implements symbolic write access with automatic (dynamic) value marshalling. The advantage of the symbolic access is (in contrast to the handle access classes SumHandleRead [▶ 902],SumHandleWrite [▶ 905]) that all type information is available when using this ADS Sum Command. The disadvantage is, that the Symbolic information must be loaded beforehand, see examples.

Examples

Usage of SumSymbolRead/SumSymbolWrite with AdsSession

```
/// <summary>
/// Defines the entry point of the application.
/// </summary>
/// <param name="args">The arguments.</param>
static void Main(string[] args)
{
    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for start:");
    Console.ReadLine();
```

```
    // Parse the command-line arguments
    AmsAddress address = ArgParser.Parse(args);

    using (AdsSession session = new AdsSession(address))
    {
    // Connect to the device target.
    AdsConnection connection = (AdsConnection)session.Connect();

    // Load symbolic information
    ReadOnlySymbolCollection symbols = session.SymbolServer.Symbols;

    ISymbol bVar1 = symbols["GVL.bVar1"];
    ISymbol bVar2 = symbols["GVL.bVar2"];
    ISymbol projectName = symbols["TwinCAT_SystemInfoVarList._AppInfo.ProjectName"];

    SymbolCollection coll = new SymbolCollection() {bVar1, bVar2, projectName};

    // Sum Command Read
    SumSymbolRead readCommand = new SumSymbolRead(connection,coll);
    object[] values = readCommand.Read();

    for (int i = 0; i < coll.Count; i++)
    {
        Console.WriteLine("Symbol: {0} (Value: {1}, Type: {2})",coll[i].InstancePath,values[i].ToStr
ing(),values[i].GetType().Name);
    }

    // Sum Command Write
    SumSymbolWrite writeCommand = new SumSymbolWrite(connection,coll);
    object[] writeValues = new object[] {true, (short) 42, "MyNewProjectName"};

    writeCommand.Write(writeValues);
    }

    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for leave:");
    Console.ReadLine();
}
```

Examples

Usage of SumSymbolRead/SumSymbolWrite with TcAdsClient

```
/// <summary>
/// Defines the entry point of the application.
/// </summary>
/// <param name="args">The arguments.</param>
static void Main(string[] args)
{
    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for start:");
    Console.ReadLine();

    // Parse the command-line arguments
    AmsAddress address = ArgParser.Parse(args);

    using (TcAdsClient client = new TcAdsClient())
    {
    // Connect the AdsClient to the device target.
    client.Connect(address);

    // Load symbolic information
    ISymbolLoader loader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    ReadOnlySymbolCollection allSymbols = loader.Symbols;

    ISymbol bVar1 = allSymbols["GVL.bVar1"];
    ISymbol bVar2 = allSymbols["GVL.iCount"];
    ISymbol projectName = allSymbols["TwinCAT_SystemInfoVarList._AppInfo.ProjectName"];

    SymbolCollection symbols = new SymbolCollection() {bVar1, bVar2, projectName};

    // Sum Command Read
    SumSymbolRead readCommand = new SumSymbolRead(client,symbols);
    object[] values = readCommand.Read();

    for (int i = 0; i < symbols.Count; i++)
    {
```

```
        Console.WriteLine("Symbol: {0} (Value: {1}, Type: {2})",symbols[i].InstancePath,values[i].To
String(),values[i].GetType().Name);
    }

    // Sum Command Write
    SumSymbolWrite writeCommand = new SumSymbolWrite(client,symbols);
    object[] writeValues = new object[] {true, (short) 42, "MyNewProjectName"};

    writeCommand.Write(writeValues);
    }

    Console.WriteLine("");
    Console.WriteLine("Press [Enter] for leave:");
    Console.ReadLine();
}
```

Reference

TwinCAT.Ads.SumCommand Namespace [▶ 892]

TwinCAT.Ads.SumCommand.ISumCommand [▶ 893]

TwinCAT.Ads.SumCommand.SumSymbolWrite

5.4.7.1 SumSymbolWrite Constructor

Initializes a new instance of the SumSymbolWrite [▶ 918] class.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SumSymbolWrite(
    IAdsConnection connection,
    IList<ISymbol> symbols
)
```

VB

```
Public Sub New (
    connection As IAdsConnection,
    symbols As IList(Of ISymbol)
)
```

Parameters

connection	Type: TwinCAT.Ads.IAdsConnection [▶ 469] The TcAdsClient or ADS Connection object
symbols	Type: System.Collections.Generic.IList.ISymbol [▶ 1634]. The symbols to read

Reference

SumSymbolWrite Class [▶ 918]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.7.2 SumSymbolWrite Methods

The SumSymbolWrite [▶ 918] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryWrite [▶ 922]	Tries the write.
	Write [▶ 923]	Writes the specified values.

Reference

SumSymbolWrite Class [▶ 918]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.7.2.1 SumSymbolWrite.TryWrite Method

Tries the write.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AdsErrorCode TryWrite(
    Object[] values,
    out AdsErrorCode[] returnCodes
)
```

VB

```
Public Function TryWrite (
    values As Object(),
    <OutAttribute> ByRef returnCodes As AdsErrorCode()
) As AdsErrorCode
```

Parameters

values	Type: .System.Object. The values.
returnCodes	Type: .TwinCAT.Ads.AdsErrorCode [▶ 305].. The return codes.

Return Value

Type: AdsErrorCode [▶ 305]
AdsErrorCode.

Remarks

The written values will be marshalled automatically to their appropriate ADS types.

Reference

SumSymbolWrite Class [▶ 918]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.4.7.2.2 SumSymbolWrite.Write Method

Writes the specified values.

Namespace: TwinCAT.Ads.SumCommand [▶ 892]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write(
    Object[] values
)
```

VB

```
Public Sub Write (
    values As Object()
)
```

Parameters

values	Type: .System.Object. The values.

Exceptions

Exception	Condition
AdsSumCommandException [▶ 384]	SumSymbolWrite failed!

Remarks

The values will be marshalled automatically to their appropriate ADS types.

Reference

SumSymbolWrite Class [▶ 918]

TwinCAT.Ads.SumCommand Namespace [▶ 892]

5.5 TwinCAT.Ads.TypeSystem Namespace

Root namespace for the ADS type system.

Classes

	Class	Description
	AliasType [▶ 925]	Alias DataType
	ArrayType [▶ 931]	Represents an Array DataType
	BitMappingType [▶ 941]	Helper Data Type to implement Bit mapping types.
	DataType [▶ 945]	DataType class
	EnumType.T. [▶ 961]	Enum DataType [▶ 945].
	Field [▶ 971]	Represents a field of an Struct/Alias/Union
	Instance [▶ 982]	Instance implementation
	Member [▶ 1001]	Represents a member of an StructType [▶ 1044]
	PointerType [▶ 1007]	Represents a pointer type.
	PrimitiveType [▶ 1010]	Class PrimitiveType.
	ReferenceType [▶ 1014]	Represents a reference type
	RpcMethod [▶ 1021]	RPC Method Description
	RpcMethodParameter [▶ 1028]	Class RpcMethodParameter.
	RpcStructType [▶ 1034]	StructType which is callable by RPC Methods.
	StringType [▶ 1039]	String DataType
	StructType [▶ 1044]	Represents a struct type
	SubRangeType.T. [▶ 1051]	Represents a SubRangType
	Symbol [▶ 1057]	Symbol class
	SymbolLoaderFactory [▶ 1100]	The class SymbolLoaderFactory [▶ 1100] is used to create a new instance of the AdsSymbolLoader initialized to the parametrized mode (SymbolBrowser V2, new Version)
	UnionType [▶ 1111]	Represents a union type
	WStringType [▶ 1114]	Represents an Unicode string (Wide string)

Interfaces

	Interface	Description
	IAdsSymbol [▶ 976]	Interface IAdsSymbol
	IAdsSymbolLoader [▶ 979]	Symbol Loader interface

5.5.1 AliasType Class

Alias DataType

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.AliasType

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class AliasType : DataType,
    IAliasType, IDataType, IBitSize
```

VB

```
Public NotInheritable Class AliasType
    Inherits DataType
    Implements IAliasType, IDataType, IBitSize
```

The AliasType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BaseType [▶ 927]	Gets the Base Type
	BaseTypeName [▶ 928]	Gets the BaseType name
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 928]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Overrides DataType.IsContainer [▶ 954].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)

	Name	Description
	IsPrimitive [▶ 929]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Overrides DataType.IsPrimitive [▶ 955].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 930]	Gets the corresponding .NET Type if attached. (Overrides DataType.ManagedType [▶ 956].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 931]	Returns a String that represents this instance. (Overrides DataType.ToString. [▶ 960].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.1.1 AliasType Properties

The AliasType [▶ 925] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BaseType [▶ 927]	Gets the Base Type
	BaseTypeName [▶ 928]	Gets the BaseType name
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)

	Name	Description
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 928]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Overrides DataType.IsContainer [▶ 954].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 929]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Overrides DataType.IsPrimitive [▶ 955].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 930]	Gets the corresponding .NET Type if attached. (Overrides DataType.ManagedType [▶ 956].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Reference

AliasType Class [▶ 925]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.1.1.1 AliasType.BaseType Property

Gets the Base Type

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType BaseType { get; }
```

VB

```
Public ReadOnly Property BaseType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]

Implements

IAliasType.BaseType [▶ 1495]

Reference

AliasType Class [▶ 925]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.1.1.2 AliasType.BaseTypeName Property

Gets the BaseType name

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string BaseTypeName { get; }
```

VB

```
Public ReadOnly Property BaseTypeName As String
    Get
```

Property Value

Type: String

Implements

IAliasType.BaseTypeName [▶ 1496]

Reference

AliasType Class [▶ 925]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.1.1.3 AliasType.IsContainer Property

Gets a value indicating whether this IDataType [▶ 1517] is a container type

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool IsContainer { get; }
```

VB

```
Public Overrides ReadOnly Property IsContainer As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is container type; otherwise, false.

Implements

IDataType.IsContainer [▶ 1522]
IDataType.IsContainer [▶ 1522]

Remarks

Container Types are all types that contain SubElements like

- Array [▶ 1189]
- Pointer [▶ 1189]
- Union [▶ 1189]
- Struct [▶ 1189]
- Function [▶ 1189]
- FunctionBlock [▶ 1189]
- Program [▶ 1189]

And the Alias [▶ 1189] types, if they have a container type as base type.

Reference

AliasType Class [▶ 925]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

IDataType.Category [▶ 1520]

5.5.1.1.4 AliasType.IsPrimitive Property

Gets a value indicating whether this IDataType [▶ 1517] is primitive

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool IsPrimitive { get; }
```

VB

```
Public Overrides ReadOnly Property IsPrimitive As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is primitive; otherwise, false.

Implements

IDataType.IsPrimitive [▶ 1523]
IDataType.IsPrimitive [▶ 1523]

Reference

AliasType Class [▶ 925]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.1.1.5 AliasType.ManagedType Property

Gets the corresponding .NET Type if attached.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override Type ManagedType { get; }
```

VB

```
Public Overrides ReadOnly Property ManagedType As Type
    Get
```

Property Value

Type: Type
Dot net type.

Reference

AliasType Class [▶ 925]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.1.2 AliasType Methods

The AliasType [▶ 925] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 931]	Returns a String that represents this instance. (Overrides DataType.ToString. [▶ 960].)

Reference

AliasType Class [▶ 925]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.1.2.1 AliasType.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

AliasType Class [▶ 925]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2 ArrayType Class

Represents an Array DataType

DataType class

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.ArrayType
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ArrayType : DataType, IArrayType,
    IDataType, IBitSize
```

VB

```
Public Class ArrayType
    Inherits DataType
    Implements IArrayType, IDataType, IBitSize
```

The ArrayType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	DimensionCount [▶ 934]	Gets the dimension count.
	Dimensions [▶ 935]	Gets the dimensions as read only collection.
	ElementCount [▶ 935]	Gets the element count.
	ElementSize [▶ 936]	Gets the byte-size of a single element of the array
	ElementType [▶ 936]	Gets the type of the contained elements.
	ElementTypeName [▶ 937]	Gets the name of the element type.
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsJagged [▶ 937]	Gets a value indicating whether this instance is jagged.
	IsOversampled [▶ 938]	Gets a value indicating whether this array instance describes an oversampling type.
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 939]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Overrides DataType.IsPrimitive [▶ 955].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	JaggedLevel [▶ 939]	Gets the jagged level (Non-Jagged Array have level 1)
	ManagedType [▶ 940]	Gets the corresponding .NET Type if attached. (Overrides DataType.ManagedType [▶ 956].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)

	Name	Description
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Fields

	Name	Description
	_dimensions [▶ 941]	Dimension information (for arrays)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.1 ArrayType Properties

The ArrayType [▶ 931] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	DimensionCount [▶ 934]	Gets the dimension count.
	Dimensions [▶ 935]	Gets the dimensions as read only collection.

	Name	Description
	ElementCount [▶ 935]	Gets the element count.
	ElementSize [▶ 936]	Gets the byte-size of a single element of the array
	ElementType [▶ 936]	Gets the type of the contained elements.
	ElementTypeName [▶ 937]	Gets the name of the element type.
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsJagged [▶ 937]	Gets a value indicating whether this instance is jagged.
	IsOversampled [▶ 938]	Gets a value indicating whether this array instance describes an oversampling type.
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 939]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Overrides DataType.IsPrimitive [▶ 955].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	JaggedLevel [▶ 939]	Gets the jagged level (Non-Jagged Array have level 1)
	ManagedType [▶ 940]	Gets the corresponding .NET Type if attached. (Overrides DataType.ManagedType [▶ 956].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.1.1 ArrayType.DimensionCount Property

Gets the dimension count.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int DimensionCount { get; }
```

VB

```
Public ReadOnly Property DimensionCount As Integer
    Get
```

Property Value

Type: Int32
The dimension count.

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.1.2 ArrayType.Dimensions Property

Gets the dimensions as read only collection.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyDimensionCollection Dimensions { get; }
```

VB

```
Public ReadOnly Property Dimensions As ReadOnlyDimensionCollection
    Get
```

Property Value

Type: ReadOnlyDimensionCollection [▶ 1731]
The dimensions.

Implements

IArrayType.Dimensions [▶ 1506]

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.1.3 ArrayType.ElementCount Property

Gets the element count.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ElementCount { get; }
```

VB

```
Public ReadOnly Property ElementCount As Integer
    Get
```

Property Value

Type: Int32
The element count.

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.1.4 ArrayType.ElementSize Property

Gets the byte-size of a single element of the array

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ElementSize { get; }
```

VB

```
Public ReadOnly Property ElementSize As Integer
    Get
```

Property Value

Type: Int32
The size of the element.

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.1.5 ArrayType.ElementType Property

Gets the type of the contained elements.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType ElementType { get; }
```

VB

```
Public ReadOnly Property ElementType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the element.

Implements

IArrayType.ElementType [▶ 1506]

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.1.6 ArrayType.ElementTypeName Property

Gets the name of the element type.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ElementTypeName { get; }
```

VB

```
Public ReadOnly Property ElementTypeName As String
    Get
```

Property Value

Type: String
The name of the element type.

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.1.7 ArrayType.IsJagged Property

Gets a value indicating whether this instance is jagged.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsJagged { get; }
```

VB

```
Public ReadOnly Property IsJagged As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is jagged; otherwise, false.

Implements

IArrayType.IsJagged [▶ 1507]

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.1.8 ArrayType.IsOversampled Property

Gets a value indicating whether this array instance describes an oversampling type.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsOversampled { get; }
```

VB

```
Public ReadOnly Property IsOversampled As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is oversampling; otherwise, false.

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.1.9 ArrayType.IsPrimitive Property

Gets a value indicating whether this IDataType [▶ 1517] is primitive

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool IsPrimitive { get; }
```

VB

```
Public Overrides ReadOnly Property IsPrimitive As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is primitive; otherwise, false.

Implements

IDataType.IsPrimitive [▶ 1523]
IDataType.IsPrimitive [▶ 1523]

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.1.10 ArrayType.JaggedLevel Property

Gets the jagged level (Non-Jagged Array have level 1)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int JaggedLevel { get; }
```

VB

```
Public ReadOnly Property JaggedLevel As Integer
    Get
```

Property Value

Type: Int32
The jagged level.

Implements

IArrayType.JaggedLevel [▶ 1507]

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.1.11 ArrayType.ManagedType Property

Gets the corresponding .NET Type if attached.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override Type ManagedType { get; }
```

VB

```
Public Overrides ReadOnly Property ManagedType As Type
    Get
```

Property Value

Type: Type
Dot net type.

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.2 ArrayType Methods

The ArrayType [▶ 931] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.3 ArrayType Fields

The ArrayType [▶ 931] type exposes the following members.

Fields

	Name	Description
	_dimensions [▶ 941]	Dimension information (for arrays)

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.2.3.1 ArrayType._dimensions Field

Dimension information (for arrays)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected DimensionCollection _dimensions
```

VB

```
Protected _dimensions As DimensionCollection
```

Field Value

Type: DimensionCollection [▶ 1209]

Reference

ArrayType Class [▶ 931]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.3 BitMappingType Class

Helper Data Type to implement Bit mapping types.

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.BitMappingType
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class BitMappingType : DataType
```

VB

```
Public NotInheritable Class BitMappingType
    Inherits DataType
```

The BitMappingType type exposes the following members.

Constructors

	Name	Description
	BitMappingType [▶ 943]	Initializes a new instance of the BitMappingType class.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)

	Name	Description
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.3.1 BitMappingType Constructor

Initializes a new instance of the BitMappingType [▶ 941] class.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public BitMappingType(
    string name,
    int bitSize,
    Type dotnetType
)
```

VB

```
Public Sub New (
    name As String,
    bitSize As Integer,
    dotnetType As Type
)
```

Parameters

name	Type: System.String The name.
bitSize	Type: System.Int32 The size of the type in bits.
dotnetType	Type: System.Type Type of the dotnet.

Reference

BitMappingType Class [▶ 941]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.3.2 BitMappingType Properties

The BitMappingType [▶ 941] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Reference

BitMappingType Class [▶ 941]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.3.3 BitMappingType Methods

The BitMappingType [▶ 941] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

BitMappingType Class [▶ 941]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4 DataType Class

DataType class

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType
    More... [▶ 947]
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class DataType : IDataType, IBitSize
```

VB

```
Public Class DataType
    Implements IDataType, IBitSize
```

The DataType type exposes the following members.

Constructors

	Name	Description
	DataType [▶ 947]	Initializes a new instance of the DataType class (copy Constructor)

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517]
	BitSize [▶ 949]	Gets the size of the DataType in bits.
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes
	Category [▶ 950]	Gets the Data Type category

	Name	Description
	Comment [▶ 951]	Gets the comment.
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name)
	Id [▶ 952]	Gets the ID of the DataType
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached.
	Name [▶ 957]	Gets the name of the Data Type (without namespace)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists.
	Size [▶ 958]	Gets the Size of the DataType in Bytes

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
S	IsPointerType [▶ 959]	Determines whether the specified category is a pointer type.
S	IsReferenceType [▶ 960]	Determines whether the specified category is a reference type.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Overrides Object.ToString..)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType
    TwinCAT.Ads.TypeSystem.AliasType [▶ 925]
    TwinCAT.Ads.TypeSystem.ArrayType [▶ 931]
    TwinCAT.Ads.TypeSystem.BitMappingType [▶ 941]
    TwinCAT.Ads.TypeSystem.EnumType.T. [▶ 961]
    TwinCAT.Ads.TypeSystem.PointerType [▶ 1007]
    TwinCAT.Ads.TypeSystem.PrimitiveType [▶ 1010]
    TwinCAT.Ads.TypeSystem.ReferenceType [▶ 1014]
    TwinCAT.Ads.TypeSystem.StringType [▶ 1039]
    TwinCAT.Ads.TypeSystem.StructType [▶ 1044]
    TwinCAT.Ads.TypeSystem.SubRangeType.T. [▶ 1051]
    TwinCAT.Ads.TypeSystem.UnionType [▶ 1111]
    TwinCAT.Ads.TypeSystem.WStringType [▶ 1114]

5.5.4.1 DataType Constructor

Initializes a new instance of the DataType [▶ 945] class (copy Constructor)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected DataType(
    DataType copy
)
```

VB

```
Protected Sub New (
    copy As DataType
)
```

Parameters

copy	Type: TwinCAT.Ads.TypeSystem.DataType [▶ 945] The copy.

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2 DataType Properties

The DataType [▶ 945] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517]

	Name	Description
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits.
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes
	Category [▶ 950]	Gets the Data Type category
	Comment [▶ 951]	Gets the comment.
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name)
	Id [▶ 952]	Gets the ID of the DataType
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached.
	Name [▶ 957]	Gets the name of the Data Type (without namespace)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists.
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.1 DataType.Attributes Property

Gets the attributes of the IDataType [▶ 1517]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyTypeAttributeCollection Attributes { get; }
```

VB

```
Public ReadOnly Property Attributes As ReadOnlyTypeAttributeCollection
    Get
```

Property Value

Type: ReadOnlyTypeAttributeCollection [▶ 1776]
The attributes.

Implements

IDataType.Attributes [▶ 1519]

Exceptions

Exception	Condition
NotImplementedException	

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.2 DataType.BitSize Property

Gets the size of the DataType [▶ 945] in bits.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int BitSize { get; }
```

VB

```
Public ReadOnly Property BitSize As Integer
    Get
```

Property Value

Type: Int32
The size of the bit.

Implements

IBitSize.BitSize [▶ 1515]

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.3 DataType.ByteSize Property

Gets the (aligned) size of of the Type/Instance in Bytes

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ByteSize { get; }
```

VB

```
Public ReadOnly Property ByteSize As Integer
    Get
```

Property Value

Type: Int32
The size of the byte.

Implements

IBitSize.ByteSize [▶ 1516]

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.4 DataType.Category Property

Gets the Data Type category

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeCategory Category { get; }
```

VB

```
Public ReadOnly Property Category As DataTypeCategory
    Get
```

Property Value

Type: DataTypeCategory [▶ 1189]
The category.

Implements

IDataType.Category [▶ 1520]

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.5 DataType.Comment Property

Gets the comment.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Comment { get; }
```

VB

```
Public ReadOnly Property Comment As String
    Get
```

Property Value

Type: String
The comment.

Implements

IDataType.Comment [▶ 1520]

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.6 DataType.FullName Property

Gets the full name of the IDataType [▶ 1517] (Namespace + Name)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string FullName { get; }
```

VB

```
Public ReadOnly Property FullName As String
    Get
```

Property Value

Type: String
The full name.

Implements

IDataType.FullName [▶ 1521]

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.7 DataType.Id Property

Gets the ID of the DataType

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Id { get; }
```

VB

```
Public ReadOnly Property Id As Integer
    Get
```

Property Value

Type: Int32
The id.

Implements

IDataType.Id [▶ 1521]

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.8 DataType.IsBitType Property

Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsBitType { get; }
```

VB

```
Public ReadOnly Property IsBitType As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is bit mapping subtype; otherwise, false.

Implements

IBitSize.IsBitType [▶ 1516]

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.9 DataType.IsByteAligned Property

Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsByteAligned { get; }
```

VB

```
Public ReadOnly Property IsByteAligned As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is byte aligned; otherwise, false.

Implements

IBitSize.IsByteAligned [▶ 1517]

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.10 DataType.IsContainer Property

Gets a value indicating whether this IDataType [▶ 1517] is a container type

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual bool IsContainer { get; }
```

VB

```
Public Overridable ReadOnly Property IsContainer As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is container type; otherwise, false.

Implements

IDataType.IsContainer [▶ 1522]

Remarks

Container Types are all types that contain SubElements like

- Array [▶ 1189]
- Pointer [▶ 1189]
- Union [▶ 1189]
- Struct [▶ 1189]
- Function [▶ 1189]
- FunctionBlock [▶ 1189]
- Program [▶ 1189]

and the Alias [▶ 1189] and Reference [▶ 1189] types, if they have a container type as base type.

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

IDataType.Category [▶ 1520]

5.5.4.2.11 DataType.IsPointer Property

Gets a value indicating whether this IDataType [▶ 1517] is a pointer type

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual bool IsPointer { get; }
```

VB

```
Public Overridable ReadOnly Property IsPointer As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is pointer type; otherwise, false.

Implements

IDataType.IsPointer [▶ 1523]

Remarks

Pointer types can be dereferenced with the '^' operator.

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

IDataType.Category [▶ 1520]

5.5.4.2.12 DataType.IsPrimitive Property

Gets a value indicating whether this IDataType [▶ 1517] is primitive

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual bool IsPrimitive { get; }
```

VB

```
Public Overridable ReadOnly Property IsPrimitive As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is primitive; otherwise, false.

Implements

IDataType.IsPrimitive [▶ 1523]

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.13 DataType.IsReference Property

Gets a value indicating whether this IDataType [▶ 1517] is a reference type

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual bool IsReference { get; }
```

VB

```
Public Overridable ReadOnly Property IsReference As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is container type; otherwise, false.

Implements

IDataType.IsReference [▶ 1524]

Remarks

Reference types can be dereferenced.

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

IDataType.Category [▶ 1520]

5.5.4.2.14 DataType.ManagedType Property

Gets the corresponding .NET Type if attached.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual Type ManagedType { get; }
```

VB

```
Public Overridable ReadOnly Property ManagedType As Type
    Get
```

Property Value

Type: Type
Dot net type.

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.15 DataType.Name Property

Gets the name of the Data Type (without namespace)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Name { get; }
```

VB

```
Public ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Implements

IDataType.Name [▶ 1524]

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.16 DataType.Namespace Property

Gets the namespace string within the IDataType [▶ 1517] exists.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Namespace { get; }
```

VB

```
Public ReadOnly Property Namespace As String
    Get
```

Property Value

Type: String
The namespace.

Implements

IDataType.Namespace [▶ 1525]

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.2.17 DataType.Size Property

Gets the Size of the DataType [▶ 945] in Bytes

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Size { get; }
```

VB

```
Public ReadOnly Property Size As Integer
    Get
```

Property Value

Type: Int32
The size.

Implements

IBitSize.Size [▶ 1517]

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.3 DataType Methods

The DataType [▶ 945] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
S	IsPointerType [▶ 959]	Determines whether the specified category is a pointer type.
S	IsReferenceType [▶ 960]	Determines whether the specified category is a reference type.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Overrides Object.ToString..)

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.3.1 DataType.IsPointerType Method

Determines whether the specified category is a pointer type.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool IsPointerType(
    DataTypeCategory cat
)
```

VB

```
Public Shared Function IsPointerType (
    cat As DataTypeCategory
) As Boolean
```

Parameters

cat	Type: TwinCAT.TypeSystem.DataTypeCategory [▶ 1189] The data type category.

Return Value

Type: Boolean
true if [is pointer type] [the specified cat]; otherwise, false.

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.3.2 DataType.IsReferenceType Method

Determines whether the specified category is a reference type.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool IsReferenceType(
    DataTypeCategory cat
)
```

VB

```
Public Shared Function IsReferenceType (
    cat As DataTypeCategory
) As Boolean
```

Parameters

cat	Type: TwinCAT.TypeSystem.DataTypeCategory [▶ 1189] The data type category.

Return Value

Type: Boolean
true if [is reference type] [the specified cat]; otherwise, false.

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.4.3.3 DataType.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

DataType Class [▶ 945]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.5 EnumType.T. Class

Enum DataType [▶ 945].

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.EnumType.T.
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class EnumType<T> : DataType,
    IEnumType<T>, IAliasType, IDataType, IBitSize, IEnumType
where T : IConvertible
```

VB

```
Public NotInheritable Class EnumType(Of T As IConvertible)
    Inherits DataType
    Implements IEnumType(Of T), IAliasType, IDataType,
    IBitSize, IEnumType
```

Type Parameters

T

The EnumType.T. type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BaseType [▶ 964]	Gets the Base Type
	BaseTypeName [▶ 964]	Gets the BaseType name
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)

	Name	Description
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	EnumValues [▶ 965]	Enumeration specification (if enum)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Methods

	Name	Description
	Contains [▶ 966]	Determines whether the enum values contains the specified name
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetNames [▶ 967]	Gets the filed names of the IEnumType.T. [▶ 1543]
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	GetValues [▶ 967]	Gets the values of the IEnumType.T. [▶ 1543]
	Parse [▶ 968]	Parses a name of the IEnumType.T. [▶ 1543] and returns the value (as base type)
	ToString. [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

	Name	Description
	ToString(Object) [▶ 969]	Returns a String that represents this instance.
	ToString(T) [▶ 970]	Returns a String that represents this instance.
	TryParse [▶ 970]	Tries to parse the Enum Value

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.5.1 EnumType.T. Properties

The EnumType.T. [▶ 961] generic type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BaseType [▶ 964]	Gets the Base Type
	BaseTypeName [▶ 964]	Gets the BaseType name
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	EnumValues [▶ 965]	Enumeration specification (if enum)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)

	Name	Description
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Reference

EnumType.T. Class [▶ 961]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.5.1.1 EnumType.T..BaseType Property

Gets the Base Type

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType BaseType { get; }
```

VB

```
Public ReadOnly Property BaseType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the base.

Implements

IAliasType.BaseType [▶ 1495]

Reference

EnumType.T. Class [▶ 961]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.5.1.2 EnumType.T..BaseTypeName Property

Gets the BaseType name

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string BaseTypeName { get; }
```

VB

```
Public ReadOnly Property BaseTypeName As String
    Get
```

Property Value

Type: String
The name of the base type.

Implements

IAliasType.BaseTypeName [▶ 1496]

Reference

EnumType.T. Class [▶ 961]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.5.1.3 EnumType.T..EnumValues Property

Enumeration specification (if enum)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyEnumValueCollection<T> EnumValues { get; }
```

VB

```
Public ReadOnly Property EnumValues As ReadOnlyEnumValueCollection(Of T)
    Get
```

Property Value

Type: ReadOnlyEnumValueCollection [▶ 1743].T [▶ 961].
The enum specification.

Implements

IEnumType.T..EnumValues [▶ 1546]

Reference

EnumType.T. Class [▶ 961]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.5.2 EnumType.T. Methods

The EnumType.T. [▶ 961] generic type exposes the following members.

Methods

	Name	Description
	Contains [▶ 966]	Determines whether the enum values contains the specified name
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetNames [▶ 967]	Gets the filed names of the IEnumType.T. [▶ 1543]
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	GetValues [▶ 967]	Gets the values of the IEnumType.T. [▶ 1543]
	Parse [▶ 968]	Parses a name of the IEnumType.T. [▶ 1543] and returns the value (as base type)
	ToString. [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)
	ToString(Object) [▶ 969]	Returns a String that represents this instance.
	ToString(T) [▶ 970]	Returns a String that represents this instance.
	TryParse [▶ 970]	Tries to parse the Enum Value

Reference

EnumType.T. Class [▶ 961]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.5.2.1 EnumType.T..Contains Method

Determines whether the enum values contains the specified name

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    string name
)
```

VB

```
Public Function Contains (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name.

Return Value

Type: Boolean
true if contains the value, otherwise, false.

Implements

IEnumType.T..Contains(String) [▶ 1547]
IEnumType.Contains(String) [▶ 1540]

Reference

EnumType.T. Class [▶ 961]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.5.2.2 EnumType.T..GetNames Method

Gets the filed names of the IEnumType.T. [▶ 1543]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string[] GetNames()
```

VB

```
Public Function GetNames As String()
```

Return Value

Type: .String.
System.String[].

Implements

IEnumType.T..GetNames. [▶ 1548]
IEnumType.GetNames. [▶ 1541]

Reference

EnumType.T. Class [▶ 961]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.5.2.3 EnumType.T..GetValues Method

Gets the values of the IEnumType.T. [▶ 1543]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T[] GetValues()
```

VB

```
Public Function GetValues As T()
```

Return Value

Type: .T [▶ 961].
T[].

Implements

IEnumType.T..GetValues. [▶ 1548]

Reference

EnumType.T. Class [▶ 961]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.5.2.4 EnumType.T..Parse Method

Parses a name of the IEnumType.T. [▶ 1543] and returns the value (as base type)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T Parse(
    string strValue
)
```

VB

```
Public Function Parse (
    strValue As String
) As T
```

Parameters

strValue	Type: System.String Enum Value as string.

Return Value

Type: T [▶ 961]
T.

Implements

IEnumType.T..Parse(String) [▶ 1548]

Reference

EnumType.T. Class [▶ 961]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.5.2.5 EnumType.T..ToString Method

Overload List

	Name	Description
	ToString. [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)
	ToString(Object) [▶ 969]	Returns a String that represents this instance.
	ToString(T) [▶ 970]	Returns a String that represents this instance.

Reference

EnumType.T. Class [▶ 961]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

EnumType.T..ToString Method (Object)

Returns a String that represents this instance.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ToString(
    Object val
)
```

VB

```
Public Function ToString (
    val As Object
) As String
```

Parameters

val	Type: System.Object The value.

Return Value

Type: String
A String that represents this instance.

Implements

IEnumType.ToString(Object) [▶ 1542]

Reference

EnumType.T. Class [▶ 961]

ToString Overload [▶ 969]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

EnumType.T..ToString Method (T)

Returns a String that represents this instance.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ToString(
    T val
)
```

VB

```
Public Function ToString (
    val As T
) As String
```

Parameters

val	Type: T [▶ 961] The value.

Return Value

Type: String
A String that represents this instance.

Implements

IEnumType.T..ToString(T) [▶ 1549]

Exceptions

Exception	Condition
ArgumentException	val

Reference

EnumType.T. Class [▶ 961]

ToString Overload [▶ 969]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.5.2.6 EnumType.T..TryParse Method

Tries to parse the Enum Value

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryParse(
    string strValue,
    out T value
)
```

VB

```
Public Function TryParse (
    strValue As String,
    <OutAttribute> ByRef value As T
) As Boolean
```

Parameters

strValue	Type: System.String Enum value (in string representation).
value	Type: T [▶ 961]. The value.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Implements

IEnumType.T..TryParse(String, T.) [▶ 1550]

Reference

EnumType.T. Class [▶ 961]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.6 Field Class

Represents a field of an Struct/Alias/Union

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.Instance [▶ 982]
    TwinCAT.Ads.TypeSystem.Field
      TwinCAT.Ads.TypeSystem.Member [▶ 1001]
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class Field : Instance, IField,
    IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Class Field
    Inherits Instance
    Implements IField, IAttributedInstance, IInstance, IBitSize
```

The Field type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 985]	Gets the Type Attributes. (Inherited from Instance [▶ 982].)
	BitSize [▶ 986]	Gets the size of this Instance [▶ 982] in bits. (Inherited from Instance [▶ 982].)
	ByteSize [▶ 986]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from Instance [▶ 982].)
	Category [▶ 987]	Gets the the DataTypeCategory [▶ 1189] of the Instance. (Inherited from Instance [▶ 982].)
	Comment [▶ 987]	Gets the comment. (Inherited from Instance [▶ 982].)
	ContextMask [▶ 988]	Gets the context mask of this instance. (Inherited from Instance [▶ 982].)
	DataType [▶ 989]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from Instance [▶ 982].)
	HasValue [▶ 989]	Gets a value indicating whether this instance has a value. (Inherited from Instance [▶ 982].)
	InstanceName [▶ 990]	Gets the name of the instance (without periods (.) (Inherited from Instance [▶ 982].)
	InstancePath [▶ 990]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from Instance [▶ 982].)
	IsBitType [▶ 991]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from Instance [▶ 982].)
	IsByteAligned [▶ 991]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from Instance [▶ 982].)
	IsPersistent [▶ 992]	Indicates if this instance is persistent. (Inherited from Instance [▶ 982].)
	IsPointer [▶ 992]	Gets a value indicating whether this instance is reference. (Inherited from Instance [▶ 982].)
	IsReadOnly [▶ 993]	Indicates if this instance is read only. (Inherited from Instance [▶ 982].)
	IsReference [▶ 993]	Gets a value indicating whether this instance is reference. (Inherited from Instance [▶ 982].)
	IsStatic [▶ 994]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from Instance [▶ 982].)
	IsTcComInterfacePointer [▶ 995]	Indicates if this instance is a TcComInterfacePointer. (Inherited from Instance [▶ 982].)
	IsTypeGuid [▶ 995]	Indicates if this instance has set TypeGuid flag. (Inherited from Instance [▶ 982].)
	Namespace [▶ 996]	Gets the namespace name. (Inherited from Instance [▶ 982].)
	ParentType [▶ 974]	Gets the Parent of this IField [▶ 1554].

	Name	Description
	Size [▶ 996]	Gets the size of the IDataType [▶ 1517] in bytes or Bits dependant on IsBitType [▶ 991] (Inherited from Instance [▶ 982].)
	TypeName [▶ 997]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from Instance [▶ 982].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnGetSize [▶ 998]	Handler function getting the size of the Instance [▶ 982] (Inherited from Instance [▶ 982].)
	OnSetInstanceName [▶ 999]	Sets a new InstanceName InstancePath (Inherited from Instance [▶ 982].)
	SetContextMask [▶ 999]	Sets the context mask. (Inherited from Instance [▶ 982].)
	ToString [▶ 1000]	Returns a String that represents this instance. (Inherited from Instance [▶ 982].)

Fields

	Name	Description
	attributes [▶ 1001]	The attributes (Inherited from Instance [▶ 982].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.6.1 Field Properties

The Field [▶ 971] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 985]	Gets the Type Attributes. (Inherited from Instance [▶ 982].)
	BitSize [▶ 986]	Gets the size of this Instance [▶ 982] in bits. (Inherited from Instance [▶ 982].)
	ByteSize [▶ 986]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from Instance [▶ 982].)
	Category [▶ 987]	Gets the the DataTypeCategory [▶ 1189] of the Instance. (Inherited from Instance [▶ 982].)

	Name	Description
	Comment [▶ 987]	Gets the comment. (Inherited from Instance [▶ 982].)
	ContextMask [▶ 988]	Gets the context mask of this instance. (Inherited from Instance [▶ 982].)
	DataType [▶ 989]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from Instance [▶ 982].)
	HasValue [▶ 989]	Gets a value indicating whether this instance has a value. (Inherited from Instance [▶ 982].)
	InstanceName [▶ 990]	Gets the name of the instance (without periods (.) (Inherited from Instance [▶ 982].)
	InstancePath [▶ 990]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from Instance [▶ 982].)
	IsBitType [▶ 991]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from Instance [▶ 982].)
	IsByteAligned [▶ 991]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from Instance [▶ 982].)
	IsPersistent [▶ 992]	Indicates if this instance is persistent. (Inherited from Instance [▶ 982].)
	IsPointer [▶ 992]	Gets a value indicating whether this instance is reference. (Inherited from Instance [▶ 982].)
	IsReadOnly [▶ 993]	Indicates if this instance is read only. (Inherited from Instance [▶ 982].)
	IsReference [▶ 993]	Gets a value indicating whether this instance is reference. (Inherited from Instance [▶ 982].)
	IsStatic [▶ 994]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from Instance [▶ 982].)
	IsTcComInterfacePointer [▶ 995]	Indicates if this instance is a TcComInterfacePointer. (Inherited from Instance [▶ 982].)
	IsTypeGuid [▶ 995]	Indicates if this instance has set TypeGuid flag. (Inherited from Instance [▶ 982].)
	Namespace [▶ 996]	Gets the namespace name. (Inherited from Instance [▶ 982].)
	ParentType [▶ 974]	Gets the Parent of this IField [▶ 1554].
	Size [▶ 996]	Gets the size of the IDataType [▶ 1517] in bytes or Bits dependant on IsBitType [▶ 991] (Inherited from Instance [▶ 982].)
	TypeName [▶ 997]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from Instance [▶ 982].)

Reference

Field Class [▶ 971]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.6.1.1 Field.ParentType Property

Gets the Parent of this IField [▶ 1554].

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType ParentType { get; }
```

VB

```
Public ReadOnly Property ParentType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the parent (Alias, Union, Struct)

Implements

IField.ParentType [▶ 1556]

Reference

Field Class [▶ 971]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.6.2 Field Methods

The Field [▶ 971] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnGetSize [▶ 998]	Handler function getting the size of the Instance [▶ 982] (Inherited from Instance [▶ 982].)
	OnSetInstanceName [▶ 999]	Sets a new InstanceName InstancePath (Inherited from Instance [▶ 982].)
	SetContextMask [▶ 999]	Sets the context mask. (Inherited from Instance [▶ 982].)
	ToString [▶ 1000]	Returns a String that represents this instance. (Inherited from Instance [▶ 982].)

Reference

Field Class [▶ 971]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.6.3 Field Fields

The Field [▶ 971] type exposes the following members.

Fields

	Name	Description
	attributes [▶ 1001]	The attributes (Inherited from Instance [▶ 982].)

Reference

Field Class [▶ 971]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.7 IAdsSymbol Interface

Interface IAdsSymbol

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IAdsSymbol : ISymbol,
    IAttributedInstance, IInstance, IBitSize, IProcessImageAddress
```

VB

```
Public Interface IAdsSymbol
    Inherits ISymbol, IAttributedInstance, IInstance, IBitSize,
    IProcessImageAddress
```

The IAdsSymbol type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)

	Name	Description
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	ImageBaseAddress [▶ 979]	Gets the AmsAddress [▶ 410] of the Process Image
	IndexGroup [▶ 1581]	Gets the index group of the Symbol (Inherited from IProcessImageAddress [▶ 1579].)
	IndexOffset [▶ 1581]	Gets the index offset of the Symbol (Inherited from IProcessImageAddress [▶ 1579].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

TwinCAT.TypeSystem.ISymbol [▶ 1634]

TwinCAT.TypeSystem.IProcessImageAddress [▶ 1579]

5.5.7.1 IAdsSymbol Properties

The IAdsSymbol [▶ 976] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	ImageBaseAddress [▶ 979]	Gets the AmsAddress [▶ 410] of the Process Image
	IndexGroup [▶ 1581]	Gets the index group of the Symbol (Inherited from IProcessImageAddress [▶ 1579].)
	IndexOffset [▶ 1581]	Gets the index offset of the Symbol (Inherited from IProcessImageAddress [▶ 1579].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)

	Name	Description
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IAdsSymbol Interface [▶ 976]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.7.1.1 IAdsSymbol.ImageBaseAddress Property

Gets the AmsAddress [▶ 410] of the Process Image

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AmsAddress ImageBaseAddress { get; }
```

VB

```
ReadOnly Property ImageBaseAddress As AmsAddress
    Get
```

Property Value

Type: AmsAddress [▶ 410]
The address.

Reference

IAdsSymbol Interface [▶ 976]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.8 IAdsSymbolLoader Interface

Symbol Loader interface

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IAdsSymbolLoader : ISymbolLoader,
    ISymbolProvider
```

VB

```
Public Interface IAdsSymbolLoader
    Inherits ISymbolLoader, ISymbolProvider
```

The IAdsSymbolLoader type exposes the following members.

Properties

	Name	Description
	BuildInTypes [▶ 1646]	Gets the build in types. (Inherited from ISymbolLoader [▶ 1645].)
	DataTypes [▶ 1648]	Gets all data types from all Namespaces (Inherited from ISymbolProvider [▶ 1647].)
	DefaultNotificationSettings [▶ 981]	Gets/Sets the default notification settings for this SymbolLoader
	ImageBaseAddress [▶ 982]	Gets the image base address.
	RootNamespaceName [▶ 1648]	Gets the name of the root namespace (Inherited from ISymbolProvider [▶ 1647].)
	Settings [▶ 1646]	Gets or sets the access Method (Inherited from ISymbolLoader [▶ 1645].)
	Symbols [▶ 1649]	Gets the (root) symbols of the Symbol provider. (Inherited from ISymbolProvider [▶ 1647].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.8.1 IAdsSymbolLoader Properties

The IAdsSymbolLoader [▶ 979] type exposes the following members.

Properties

	Name	Description
	BuildInTypes [▶ 1646]	Gets the build in types. (Inherited from ISymbolLoader [▶ 1645].)
	DataTypes [▶ 1648]	Gets all data types from all Namespaces (Inherited from ISymbolProvider [▶ 1647].)
	DefaultNotificationSettings [▶ 981]	Gets/Sets the default notification settings for this SymbolLoader
	ImageBaseAddress [▶ 982]	Gets the image base address.
	RootNamespaceName [▶ 1648]	Gets the name of the root namespace (Inherited from ISymbolProvider [▶ 1647].)
	Settings [▶ 1646]	Gets or sets the access Method (Inherited from ISymbolLoader [▶ 1645].)
	Symbols [▶ 1649]	Gets the (root) symbols of the Symbol provider. (Inherited from ISymbolProvider [▶ 1647].)

Reference

IAdsSymbolLoader Interface [▶ 979]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.8.1.1 IAdsSymbolLoader.DefaultNotificationSettings Property

Gets/Sets the default notification settings for this SymbolLoader

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
INotificationSettings DefaultNotificationSettings { get; set; }
```

VB

```
Property DefaultNotificationSettings As INotificationSettings
    Get
    Set
```

Property Value

Type: INotificationSettings [▶ 1566]
The default notification settings.

Remarks

The Default notification Settings can be set on the SymbolLoader and is used as default on the different Symbols. On the symbol itself the NotificationSettings [▶ 573] can be overridden.

Examples

Setting the DefaultNotificationSettings on the IAdsSymbolLoader [▶ 979] object:

Set DefaultNotificationSettings

```
// Create AdsClient object
using (TcAdsClient client = new TcAdsClient())
{
    // No automatic Synchronization (necessary for Console applications without message loop)
    client.Synchronize = false;

    // Connect to client
    client.Connect(address);

    // Usage of 'dynamic' type/symbol loader
    SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.DynamicTree, ValueAcces
sMode.IndexGroupOffsetPreferred);
    IAdsSymbolLoader dynLoader = (IAdsSymbolLoader)SymbolLoaderFactory.Create(client, settings);

    // Set the DefaultNotification Properties
    dynLoader.DefaultNotificationSettings = new NotificationSettings(AdsTransMode.ClientOnChange, 20
0, 2000);

    // Determine the symbols
    dynamic dynamicSymbols = ((IDynamicSymbolLoader)dynLoader).SymbolsDynamic;

    // Task 1 Symbol (build in symbol)
    dynamic task1Symbol = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo[1];

    // CycleCount Symbol
    dynamic cycleCountSymbol = task1Symbol.CycleCount;
```

```
    // Override Notification Setting for Cycle Count Symbol
    cycleCountSymbol.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 250, 0);

    // Register Dynamic Value Changed event.
    cycleCountSymbol.ValueChanged += new EventHandler<ValueChangedArgs>(cycleCount_ValueChanged);
```

Reference

IAdsSymbolLoader Interface [▶ 979]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.8.1.2 IAdsSymbolLoader.ImageBaseAddress Property

Gets the image base address.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
AmsAddress ImageBaseAddress { get; }
```

VB

```
ReadOnly Property ImageBaseAddress As AmsAddress
    Get
```

Property Value

Type: AmsAddress [▶ 410]
The image base address.

Reference

IAdsSymbolLoader Interface [▶ 979]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9 Instance Class

Instance implementation

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.Instance
    TwinCAT.Ads.TypeSystem.Field [▶ 971]
    TwinCAT.Ads.TypeSystem.Symbol [▶ 1057]
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class Instance : IInstance, IBitSize
```

VB

```
Public Class Instance
    Implements IInstance, IBitSize
```

The Instance type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 985]	Gets the Type Attributes.
	BitSize [▶ 986]	Gets the size of this Instance in bits.
	ByteSize [▶ 986]	Gets the (aligned) size of of the Type/Instance in Bytes
	Category [▶ 987]	Gets the the DataTypeCategory [▶ 1189] of the Instance.
	Comment [▶ 987]	Gets the comment.
	ContextMask [▶ 988]	Gets the context mask of this instance.
	DataType [▶ 989]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556].
	HasValue [▶ 989]	Gets a value indicating whether this instance has a value.
	InstanceName [▶ 990]	Gets the name of the instance (without periods (.)
	InstancePath [▶ 990]	Gets the relative / absolute access path to the instance (with periods (.))
	IsBitType [▶ 991]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping
	IsByteAligned [▶ 991]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0)
	IsPersistent [▶ 992]	Indicates if this instance is persistent.
	IsPointer [▶ 992]	Gets a value indicating whether this instance is reference.
	IsReadOnly [▶ 993]	Indicates if this instance is read only.
	IsReference [▶ 993]	Gets a value indicating whether this instance is reference.
	IsStatic [▶ 994]	Gets a value indicating whether this IInstance [▶ 1556] is static.
	IsTcComInterfacePointer [▶ 995]	Indicates if this instance is a TcComInterfacePointer.
	IsTypeGuid [▶ 995]	Indicates if this instance has set TypeGuid flag.
	Namespace [▶ 996]	Gets the namespace name.
	Size [▶ 996]	Gets the size of the IDataType [▶ 1517] in bytes or Bits dependant on IsBitType [▶ 991]
	TypeName [▶ 997]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556].

Methods

	Name	Description
	AlignTypeName [▶ 998]	Aligns the type name
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnGetSize [▶ 998]	Handler function getting the size of the Instance
	OnSetInstanceName [▶ 999]	Sets a new InstanceName InstancePath
	SetContextMask [▶ 999]	Sets the context mask.
	ToString [▶ 1000]	Returns a String that represents this instance. (Overrides Object.ToString..)

Fields

	Name	Description
	attributes [▶ 1001]	The attributes

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1 Instance Properties

The Instance [▶ 982] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 985]	Gets the Type Attributes.
	BitSize [▶ 986]	Gets the size of this Instance [▶ 982] in bits.
	ByteSize [▶ 986]	Gets the (aligned) size of of the Type/Instance in Bytes
	Category [▶ 987]	Gets the the DataTypeCategory [▶ 1189] of the Instance.
	Comment [▶ 987]	Gets the comment.
	ContextMask [▶ 988]	Gets the context mask of this instance.

	Name	Description
	DataType [▶ 989]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556].
	HasValue [▶ 989]	Gets a value indicating whether this instance has a value.
	InstanceName [▶ 990]	Gets the name of the instance (without periods (.)
	InstancePath [▶ 990]	Gets the relative / absolute access path to the instance (with periods (.))
	IsBitType [▶ 991]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping
	IsByteAligned [▶ 991]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0)
	IsPersistent [▶ 992]	Indicates if this instance is persistent.
	IsPointer [▶ 992]	Gets a value indicating whether this instance is reference.
	IsReadOnly [▶ 993]	Indicates if this instance is read only.
	IsReference [▶ 993]	Gets a value indicating whether this instance is reference.
	IsStatic [▶ 994]	Gets a value indicating whether this IInstance [▶ 1556] is static.
	IsTcComInterfacePointer [▶ 995]	Indicates if this instance is a TcComInterfacePointer.
	IsTypeGuid [▶ 995]	Indicates if this instance has set TypeGuid flag.
	Namespace [▶ 996]	Gets the namespace name.
	Size [▶ 996]	Gets the size of the IDataType [▶ 1517] in bytes or Bits dependant on IsBitType [▶ 991]
	TypeName [▶ 997]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556].

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.1 Instance.Attributes Property

Gets the Type Attributes.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyTypeAttributeCollection Attributes { get; }
```

VB

```
Public ReadOnly Property Attributes As ReadOnlyTypeAttributeCollection
    Get
```

Property Value

Type: ReadOnlyTypeAttributeCollection [▶ 1776]
The attributes.

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.2 Instance.BitSize Property

Gets the size of this Instance [▶ 982] in bits.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual int BitSize { get; }
```

VB

```
Public Overridable ReadOnly Property BitSize As Integer
    Get
```

Property Value

Type: Int32
The size of the bit.

Implements

IBitSize.BitSize [▶ 1515]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.3 Instance.ByteSize Property

Gets the (aligned) size of of the Type/Instance in Bytes

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ByteSize { get; }
```

VB

```
Public ReadOnly Property ByteSize As Integer
    Get
```

Property Value

Type: Int32
The size of the byte.

Implements

IBitSize.ByteSize [▶ 1516]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.4 Instance.Category Property

Gets the the DataTypeCategory [▶ 1189] of the Instance.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeCategory Category { get; }
```

VB

```
Public ReadOnly Property Category As DataTypeCategory
    Get
```

Property Value

Type: DataTypeCategory [▶ 1189]
The category.

Remarks

Corresponds to the Category [▶ 1520]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.5 Instance.Comment Property

Gets the comment.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Comment { get; }
```

VB

```
Public ReadOnly Property Comment As String
    Get
```

Property Value

Type: String
The comment.

Implements

IInstance.Comment [▶ 1558]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.6 Instance.ContextMask Property

Gets the context mask of this instance.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public byte ContextMask { get; }
```

VB

```
Public ReadOnly Property ContextMask As Byte
    Get
```

Property Value

Type: Byte

Remarks

The Size of the internal data is 4-Bit

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.7 Instance.DataType Property

Gets the IDataType [▶ 1517] of the IInstance [▶ 1556].

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType DataType { get; }
```

VB

```
Public ReadOnly Property DataType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the data.

Implements

IInstance.DataType [▶ 1559]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.8 Instance.HasValue Property

Gets a value indicating whether this instance has a value.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual bool HasValue { get; }
```

VB

```
Public Overridable ReadOnly Property HasValue As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has value; otherwise, false.

Remarks

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.9 Instance.InstanceName Property

Gets the name of the instance (without periods (.)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string InstanceName { get; }
```

VB

```
Public ReadOnly Property InstanceName As String
    Get
```

Property Value

Type: String
The name of the instance.

Implements

IInstance.InstanceName [▶ 1559]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.10 Instance.InstancePath Property

Gets the relative / absolute access path to the instance (with periods (.))

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual string InstancePath { get; }
```

VB

```
Public Overridable ReadOnly Property InstancePath As String
    Get
```

Property Value

Type: String
The instance path.

Implements

IInstance.InstancePath [▶ 1560]

Remarks

If this path is relative or absolute depends on the context. IMember [▶ 1562] are using relative paths, ISymbol [▶ 1634]s are using absolute ones.

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.11 Instance.IsBitType Property

Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsBitType { get; }
```

VB

```
Public ReadOnly Property IsBitType As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is bit mapping; otherwise, false.

Implements

IBitSize.IsBitType [▶ 1516]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.12 Instance.IsByteAligned Property

Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsByteAligned { get; }
```

VB

```
Public ReadOnly Property IsByteAligned As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is byte aligned; otherwise, false.

Implements

IBitSize.IsByteAligned [▶ 1517]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.13 Instance.IsPersistent Property

Indicates if this instance is persistent.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsPersistent { get; }
```

VB

```
Public ReadOnly Property IsPersistent As Boolean
    Get
```

Property Value

Type: Boolean

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.14 Instance.IsPointer Property

Gets a value indicating whether this instance is reference.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsPointer { get; }
```

VB

```
Public ReadOnly Property IsPointer As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is reference; otherwise, false.

Implements

IInstance.IsPointer [▶ 1560]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.15 Instance.IsReadOnly Property

Indicates if this instance is read only.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.16 Instance.IsReference Property

Gets a value indicating whether this instance is reference.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReference { get; }
```

VB

```
Public ReadOnly Property IsReference As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is reference; otherwise, false.

Implements

IInstance.IsReference [▶ 1561]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.17 Instance.IsStatic Property

Gets a value indicating whether this IInstance [▶ 1556] is static.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsStatic { get; }
```

VB

```
Public ReadOnly Property IsStatic As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is static; otherwise, false.

Implements

IInstance.IsStatic [▶ 1561]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.18 Instance.IsTcComInterfacePointer Property

Indicates if this instance is a TcComInterfacePointer.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsTcComInterfacePointer { get; }
```

VB

```
Public ReadOnly Property IsTcComInterfacePointer As Boolean
    Get
```

Property Value

Type: Boolean

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.19 Instance.IsTypeGuid Property

Indicates if this instance has set TypeGuid flag.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsTypeGuid { get; }
```

VB

```
Public ReadOnly Property IsTypeGuid As Boolean
    Get
```

Property Value

Type: Boolean

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.20 Instance.Namespace Property

Gets the namespace name.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Namespace { get; }
```

VB

```
Public ReadOnly Property Namespace As String
    Get
```

Property Value

Type: String
The namespace.

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.21 Instance.Size Property

Gets the size of the IDataType [▶ 1517] in bytes or Bits dependant on IsBitType [▶ 991]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Size { get; }
```

VB

```
Public ReadOnly Property Size As Integer
    Get
```

Property Value

Type: Int32
The size of the bit.

Implements

IBitSize.Size [▶ 1517]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.1.22 Instance.TypeName Property

Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556].

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string TypeName { get; }
```

VB

```
Public ReadOnly Property TypeName As String
    Get
```

Property Value

Type: String
The name of the type.

Implements

IInstance.TypeName [▶ 1562]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.2 Instance Methods

The Instance [▶ 982] type exposes the following members.

Methods

	Name	Description
	AlignTypeName [▶ 998]	Aligns the type name
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnGetSize [▶ 998]	Handler function getting the size of the Instance [▶ 982]
	OnSetInstanceName [▶ 999]	Sets a new InstanceName InstancePath

	Name	Description
	SetContextMask [▶ 999]	Sets the context mask.
	ToString [▶ 1000]	Returns a String that represents this instance. (Overrides Object.ToString..)

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.2.1 Instance.AlignTypeName Method

Aligns the type name

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected static string AlignTypeName(
    string typeName
)
```

VB

```
Protected Shared Function AlignTypeName (
    typeName As String
) As String
```

Parameters

typeName	Type: System.String Name of the type.

Return Value

Type: String
System.String.

Exceptions

Exception	Condition
ArgumentException	Type name not valid!

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.2.2 Instance.OnGetSize Method

Handler function getting the size of the Instance [▶ 982]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual int OnGetSize()
```

VB

```
Protected Overridable Function OnGetSize As Integer
```

Return Value

Type: Int32
System.Int32.

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.2.3 Instance.OnSetInstanceName Method

Sets a new InstanceName InstancePath

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual void OnSetInstanceName(
    string instanceName
)
```

VB

```
Protected Overridable Sub OnSetInstanceName (
    instanceName As String
)
```

Parameters

instanceName	Type: System.String Instance name.

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.2.4 Instance.SetContextMask Method

Sets the context mask.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected void SetContextMask(
    byte contextMask
)
```

VB

```
Protected Sub SetContextMask (
    contextMask As Byte
)
```

Parameters

contextMask	Type: System.Byte The context mask.

Exceptions

Exception	Condition
ArgumentOutOfRangeException	contextMask

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.2.5 Instance.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.3 Instance Fields

The Instance [▶ 982] type exposes the following members.

Fields

	Name	Description
	attributes [▶ 1001]	The attributes

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.9.3.1 Instance.attributes Field

The attributes

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected TypeAttributeCollection attributes
```

VB

```
Protected attributes As TypeAttributeCollection
```

Field Value

Type: TypeAttributeCollection [▶ 1829]

Reference

Instance Class [▶ 982]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.10 Member Class

Represents a member of an StructType [▶ 1044]

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.Instance [▶ 982]
    TwinCAT.Ads.TypeSystem.Field [▶ 971]
      TwinCAT.Ads.TypeSystem.Member
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class Member : Field,
    IMember, IField, IAttributedInstance, IInstance, IBitSize
```

VB

```
Public NotInheritable Class Member
    Inherits Field
    Implements IMember, IField, IAttributedInstance, IInstance,
    IBitSize
```

The Member type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 985]	Gets the Type Attributes. (Inherited from Instance [▶ 982].)
	BitOffset [▶ 1004]	Gets the bit offset.
	BitSize [▶ 986]	Gets the size of this Instance [▶ 982] in bits. (Inherited from Instance [▶ 982].)
	ByteOffset [▶ 1005]	Gets the byte offset.
	ByteSize [▶ 986]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from Instance [▶ 982].)
	Category [▶ 987]	Gets the the DataTypeCategory [▶ 1189] of the Instance. (Inherited from Instance [▶ 982].)
	Comment [▶ 987]	Gets the comment. (Inherited from Instance [▶ 982].)
	ContextMask [▶ 988]	Gets the context mask of this instance. (Inherited from Instance [▶ 982].)
	DataType [▶ 989]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from Instance [▶ 982].)
	HasValue [▶ 989]	Gets a value indicating whether this instance has a value. (Inherited from Instance [▶ 982].)
	InstanceName [▶ 990]	Gets the name of the instance (without periods (.) (Inherited from Instance [▶ 982].)
	InstancePath [▶ 990]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from Instance [▶ 982].)
	IsBitType [▶ 991]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from Instance [▶ 982].)
	IsByteAligned [▶ 991]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from Instance [▶ 982].)
	IsPersistent [▶ 992]	Indicates if this instance is persistent. (Inherited from Instance [▶ 982].)
	IsPointer [▶ 992]	Gets a value indicating whether this instance is reference. (Inherited from Instance [▶ 982].)
	IsReadOnly [▶ 993]	Indicates if this instance is read only. (Inherited from Instance [▶ 982].)
	IsReference [▶ 993]	Gets a value indicating whether this instance is reference. (Inherited from Instance [▶ 982].)

	Name	Description
	IsStatic [▶ 994]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from Instance [▶ 982].)
	IsTcComInterfacePointer [▶ 995]	Indicates if this instance is a TcComInterfacePointer. (Inherited from Instance [▶ 982].)
	IsTypeGuid [▶ 995]	Indicates if this instance has set TypeGuid flag. (Inherited from Instance [▶ 982].)
	Namespace [▶ 996]	Gets the namespace name. (Inherited from Instance [▶ 982].)
	Offset [▶ 1006]	Gets the offset of the Member within the parent StructType [▶ 1044] in bits or bytes dependent on IsBitType [▶ 991]
	ParentType [▶ 974]	Gets the Parent of this IField [▶ 1554]. (Inherited from Field [▶ 971].)
	Size [▶ 996]	Gets the size of the IDataType [▶ 1517] in bytes or Bits dependant on IsBitType [▶ 991] (Inherited from Instance [▶ 982].)
	TypeName [▶ 997]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from Instance [▶ 982].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 1000]	Returns a String that represents this instance. (Inherited from Instance [▶ 982].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.10.1 Member Properties

The Member [▶ 1001] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 985]	Gets the Type Attributes. (Inherited from Instance [▶ 982].)
	BitOffset [▶ 1004]	Gets the bit offset.
	BitSize [▶ 986]	Gets the size of this Instance [▶ 982] in bits. (Inherited from Instance [▶ 982].)
	ByteOffset [▶ 1005]	Gets the byte offset.
	ByteSize [▶ 986]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from Instance [▶ 982].)
	Category [▶ 987]	Gets the the DataTypeCategory [▶ 1189] of the Instance. (Inherited from Instance [▶ 982].)

	Name	Description
	Comment [▶ 987]	Gets the comment. (Inherited from Instance [▶ 982].)
	ContextMask [▶ 988]	Gets the context mask of this instance. (Inherited from Instance [▶ 982].)
	DataType [▶ 989]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from Instance [▶ 982].)
	HasValue [▶ 989]	Gets a value indicating whether this instance has a value. (Inherited from Instance [▶ 982].)
	InstanceName [▶ 990]	Gets the name of the instance (without periods (.) (Inherited from Instance [▶ 982].)
	InstancePath [▶ 990]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from Instance [▶ 982].)
	IsBitType [▶ 991]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from Instance [▶ 982].)
	IsByteAligned [▶ 991]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from Instance [▶ 982].)
	IsPersistent [▶ 992]	Indicates if this instance is persistent. (Inherited from Instance [▶ 982].)
	IsPointer [▶ 992]	Gets a value indicating whether this instance is reference. (Inherited from Instance [▶ 982].)
	IsReadOnly [▶ 993]	Indicates if this instance is read only. (Inherited from Instance [▶ 982].)
	IsReference [▶ 993]	Gets a value indicating whether this instance is reference. (Inherited from Instance [▶ 982].)
	IsStatic [▶ 994]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from Instance [▶ 982].)
	IsTcComInterfacePointer [▶ 995]	Indicates if this instance is a TcComInterfacePointer. (Inherited from Instance [▶ 982].)
	IsTypeGuid [▶ 995]	Indicates if this instance has set TypeGuid flag. (Inherited from Instance [▶ 982].)
	Namespace [▶ 996]	Gets the namespace name. (Inherited from Instance [▶ 982].)
	Offset [▶ 1006]	Gets the offset of the Member [▶ 1001] within the parent StructType [▶ 1044] in bits or bytes dependent on IsBitType [▶ 991]
	ParentType [▶ 974]	Gets the Parent of this IField [▶ 1554]. (Inherited from Field [▶ 971].)
	Size [▶ 996]	Gets the size of the IDataType [▶ 1517] in bytes or Bits dependant on IsBitType [▶ 991] (Inherited from Instance [▶ 982].)
	TypeName [▶ 997]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from Instance [▶ 982].)

Reference

Member Class [▶ 1001]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.10.1.1 Member.BitOffset Property

Gets the bit offset.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int BitOffset { get; }
```

VB

```
Public ReadOnly Property BitOffset As Integer
    Get
```

Property Value

Type: Int32
The bit offset.

Implements

IMember.BitOffset [▶ 1564]

Reference

Member Class [▶ 1001]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.10.1.2 Member.ByteOffset Property

Gets the byte offset.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ByteOffset { get; }
```

VB

```
Public ReadOnly Property ByteOffset As Integer
    Get
```

Property Value

Type: Int32
The byte offset.

Implements

IMember.ByteOffset [▶ 1565]

Reference

Member Class [▶ 1001]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.10.1.3 Member.Offset Property

Gets the offset of the Member [▶ 1001] within the parent StructType [▶ 1044] in bits or bytes dependent on IsBitType [▶ 991]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Offset { get; }
```

VB

```
Public ReadOnly Property Offset As Integer
    Get
```

Property Value

Type: Int32
The offset.

Implements

IMember.Offset [▶ 1565]

Reference

Member Class [▶ 1001]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.10.2 Member Methods

The Member [▶ 1001] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 1000]	Returns a String that represents this instance. (Inherited from Instance [▶ 982].)

Reference

Member Class [▶ 1001]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.11 PointerType Class

Represents a pointer type.

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.PointerType
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class PointerType : DataType,
    IPointerType, IDataType, IBitSize
```

VB

```
Public NotInheritable Class PointerType
    Inherits DataType
    Implements IPointerType, IDataType, IBitSize
```

The PointerType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)

	Name	Description
	ManagedType [▶ 1009]	Gets the corresponding .NET Type if attached. (Overrides DataType.ManagedType [▶ 956].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	ReferencedType [▶ 1010]	Gets the the referenced type.
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.11.1 PointerType Properties

The PointerType [▶ 1007] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)

	Name	Description
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 1009]	Gets the corresponding .NET Type if attached. (Overrides DataType.ManagedType [▶ 956].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	ReferencedType [▶ 1010]	Gets the the referenced type.
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Reference

PointerType Class [▶ 1007]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.11.1.1 PointerType.ManagedType Property

Gets the corresponding .NET Type if attached.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override Type ManagedType { get; }
```

VB

```
Public Overrides ReadOnly Property ManagedType As Type
    Get
```

Property Value

Type: Type
Dot net type.

Reference

PointerType Class [▶ 1007]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.11.1.2 PointerType.ReferencedType Property

Gets the the referenced type.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType ReferencedType { get; }
```

VB

```
Public ReadOnly Property ReferencedType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the referenced.

Implements

IPointerType.ReferencedType [▶ 1576]

Reference

PointerType Class [▶ 1007]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.11.2 PointerType Methods

The PointerType [▶ 1007] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

PointerType Class [▶ 1007]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.12 PrimitiveType Class

Class PrimitiveType.

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.PrimitiveType
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class PrimitiveType : DataType,
    IPrimitiveType, IDataType, IBitSize
```

VB

```
Public NotInheritable Class PrimitiveType
    Inherits DataType
    Implements IPrimitiveType, IDataType, IBitSize
```

The PrimitiveType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)

	Name	Description
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	PrimitiveFlags [▶ 1013]	Indicates types of different PrimitiveTypes with flags.
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

TwinCAT.Ads.TypeSystem.DataType [▶ 945]

5.5.12.1 PrimitiveType Properties

The PrimitiveType [▶ 1010] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)

	Name	Description
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	PrimitiveFlags [▶ 1013]	Indicates types of different PrimitiveTypes with flags.
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Reference

PrimitiveType Class [▶ 1010]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.12.1.1 PrimitiveType.PrimitiveFlags Property

Indicates types of different PrimitiveTypes with flags.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public PrimitiveTypeFlags PrimitiveFlags { get; }
```

VB

```
Public ReadOnly Property PrimitiveFlags As PrimitiveTypeFlags
    Get
```

Property Value

Type: PrimitiveTypeFlags [▶ 1724]
The primitive flags.

Implements

IPrimitiveType.PrimitiveFlags [▶ 1579]

Reference

PrimitiveType Class [▶ 1010]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.12.2 PrimitiveType Methods

The PrimitiveType [▶ 1010] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

PrimitiveType Class [▶ 1010]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.13 ReferenceType Class

Represents a reference type

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.ReferenceType
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class ReferenceType : DataType,
    IReferenceType, IDataType, IBitSize
```

VB

```
Public NotInheritable Class ReferenceType
    Inherits DataType
    Implements IReferenceType, IDataType, IBitSize
```

The ReferenceType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)

	Name	Description
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 1017]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Overrides DataType.IsContainer [▶ 954].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 1018]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Overrides DataType.IsPrimitive [▶ 955].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 1018]	Gets the corresponding .NET Type if attached. (Overrides DataType.ManagedType [▶ 956].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	ReferencedType [▶ 1019]	Gets the the referenced type.
	ResolvedByteSize [▶ 1019]	Get the ByteSize of the (completely) resolved Symbol
	ResolvedCategory [▶ 1020]	Gets the Category of the (completely) resolved Symbol.
	ResolvedType [▶ 1020]	Gets the (completely) resolved type
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.13.1 ReferenceType Properties

The ReferenceType [▶ 1014] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 1017]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Overrides DataType.IsContainer [▶ 954].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 1018]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Overrides DataType.IsPrimitive [▶ 955].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 1018]	Gets the corresponding .NET Type if attached. (Overrides DataType.ManagedType [▶ 956].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	ReferencedType [▶ 1019]	Gets the the referenced type.
	ResolvedByteSize [▶ 1019]	Get the ByteSize of the (completely) resolved Symbol
	ResolvedCategory [▶ 1020]	Gets the Category of the (completely) resolved Symbol.
	ResolvedType [▶ 1020]	Gets the (completely) resolved type

	Name	Description
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Reference

ReferenceType Class [▶ 1014]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.13.1.1 ReferenceType.IsContainer Property

Gets a value indicating whether this IDataType [▶ 1517] is a container type

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool IsContainer { get; }
```

VB

```
Public Overrides ReadOnly Property IsContainer As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is container type; otherwise, false.

Implements

IDataType.IsContainer [▶ 1522]
IDataType.IsContainer [▶ 1522]

Remarks

Container Types are all types that contain SubElements like

- Array [▶ 1189]
- Pointer [▶ 1189]
- Union [▶ 1189]
- Struct [▶ 1189]
- Function [▶ 1189]
- FunctionBlock [▶ 1189]
- Program [▶ 1189]

and the Alias [▶ 1189] and Reference [▶ 1189] types, if they have a container type as base type.

Reference

ReferenceType Class [▶ 1014]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

IDataType.Category [▶ 1520]

5.5.13.1.2 ReferenceType.IsPrimitive Property

Gets a value indicating whether this IDataType [▶ 1517] is primitive

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool IsPrimitive { get; }
```

VB

```
Public Overrides ReadOnly Property IsPrimitive As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is primitive; otherwise, false.

Implements

IDataType.IsPrimitive [▶ 1523]
IDataType.IsPrimitive [▶ 1523]

Reference

ReferenceType Class [▶ 1014]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.13.1.3 ReferenceType.ManagedType Property

Gets the corresponding .NET Type if attached.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override Type ManagedType { get; }
```

VB

```
Public Overrides ReadOnly Property ManagedType As Type
    Get
```

Property Value

Type: Type
Dot net type.

Reference

ReferenceType Class [▶ 1014]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.13.1.4 ReferenceType.ReferencedType Property

Gets the the referenced type.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType ReferencedType { get; }
```

VB

```
Public ReadOnly Property ReferencedType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the referenced.

Implements

IReferenceType.ReferencedType [▶ 1589]

Reference

ReferenceType Class [▶ 1014]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.13.1.5 ReferenceType.ResolvedByteSize Property

Get the ByteSize of the (completely) resolved Symbol

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ResolvedByteSize { get; }
```

VB

```
Public ReadOnly Property ResolvedByteSize As Integer
    Get
```

Property Value

Type: Int32
The size of the resolved byte.

Implements

IReferenceType.ResolvedByteSize [▶ 1589]

Reference

ReferenceType Class [▶ 1014]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.13.1.6 ReferenceType.ResolvedCategory Property

Gets the Category of the (completely) resolved Symbol.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeCategory ResolvedCategory { get; }
```

VB

```
Public ReadOnly Property ResolvedCategory As DataTypeCategory
    Get
```

Property Value

Type: DataTypeCategory [▶ 1189]
The resolved category.

Implements

IReferenceType.ResolvedCategory [▶ 1590]

Reference

ReferenceType Class [▶ 1014]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.13.1.7 ReferenceType.ResolvedType Property

Gets the (completely) resolved type

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType ResolvedType { get; }
```

VB

```
Public ReadOnly Property ResolvedType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the resolved symbol

Implements

IReferenceType.ResolvedType [▶ 1590]

Reference

ReferenceType Class [▶ 1014]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.13.2 ReferenceType Methods

The ReferenceType [▶ 1014] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

ReferenceType Class [▶ 1014]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.14 RpcMethod Class

RPC Method Description

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.RpcMethod
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class RpcMethod : IRpcMethod
```

VB

```
Public Class RpcMethod
    Implements IRpcMethod
```

The RpcMethod type exposes the following members.

Properties

	Name	Description
	Comment [▶ 1023]	Gets the Method comment.
	IsVoid [▶ 1024]	Gets a value indicating whether this IRpcMethod [▶ 1597] has no return parameter
	Name [▶ 1024]	Gets the name of the method
	Parameters [▶ 1025]	Gets the Method parameter descriptions.
	ReturnAlignSize [▶ 1025]	Gets the size of the biggest element in bytes for Alignment
	ReturnType [▶ 1026]	Gets the return type.
	ReturnTypeSize [▶ 1026]	Gets the Byte size of the return type.
	VTableIndex [▶ 1027]	Gets the V-table index of the method.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString [▶ 1028]	Returns a String that represents this instance. (Overrides Object.ToString..)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.14.1 RpcMethod Properties

The RpcMethod [▶ 1021] type exposes the following members.

Properties

	Name	Description
	Comment [▶ 1023]	Gets the Method comment.
	IsVoid [▶ 1024]	Gets a value indicating whether this IRpcMethod [▶ 1597] has no return parameter
	Name [▶ 1024]	Gets the name of the method
	Parameters [▶ 1025]	Gets the Method parameter descriptions.
	ReturnAlignSize [▶ 1025]	Gets the size of the biggest element in bytes for Alignment
	ReturnType [▶ 1026]	Gets the return type.
	ReturnTypeSize [▶ 1026]	Gets the Byte size of the return type.
	VTableIndex [▶ 1027]	Gets the V-table index of the method.

Reference

RpcMethod Class [▶ 1021]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.14.1.1 RpcMethod.Comment Property

Gets the Method comment.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Comment { get; }
```

VB

```
Public ReadOnly Property Comment As String
    Get
```

Property Value

Type: String
The comment.

Implements

IRpcMethod.Comment [▶ 1598]

Reference

RpcMethod Class [▶ 1021]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.14.1.2 RpcMethod.IsVoid Property

Gets a value indicating whether this IRpcMethod [▶ 1597] has no return parameter

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsVoid { get; }
```

VB

```
Public ReadOnly Property IsVoid As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is void; otherwise, false.

Implements

IRpcMethod.IsVoid [▶ 1599]

Exceptions

Exception	Condition
NotImplementedException	

Reference

RpcMethod Class [▶ 1021]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.14.1.3 RpcMethod.Name Property

Gets the name of the method

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Name { get; }
```

VB

```
Public ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Implements

IRpcMethod.Name [▶ 1599]

Reference

RpcMethod Class [▶ 1021]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.14.1.4 RpcMethod.Parameters Property

Gets the Method parameter descriptions.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyMethodParameterCollection Parameters { get; }
```

VB

```
Public ReadOnly Property Parameters As ReadOnlyMethodParameterCollection
    Get
```

Property Value

Type: ReadOnlyMethodParameterCollection [▶ 1759]
The parameters.

Implements

IRpcMethod.Parameters [▶ 1600]

Reference

RpcMethod Class [▶ 1021]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.14.1.5 RpcMethod.ReturnAlignSize Property

Gets the size of the biggest element in bytes for Alignment

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReturnAlignSize { get; }
```

VB

```
Public ReadOnly Property ReturnAlignSize As Integer
    Get
```

Property Value

Type: Int32
The size of the return align.

Reference

RpcMethod Class [▶ 1021]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.14.1.6 RpcMethod.ReturnType Property

Gets the return type.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string ReturnType { get; }
```

VB

```
Public ReadOnly Property ReturnType As String
    Get
```

Property Value

Type: String
Return type.

Implements

IRpcMethod.ReturnType [▶ 1600]

Reference

RpcMethod Class [▶ 1021]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.14.1.7 RpcMethod.ReturnTypeSize Property

Gets the Byte size of the return type.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ReturnTypeSize { get; }
```

VB

```
Public ReadOnly Property ReturnTypeSize As Integer
    Get
```

Property Value

Type: Int32
The size of the return type.

Implements

IRpcMethod.ReturnTypeSize [▶ 1601]

Reference

RpcMethod Class [▶ 1021]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.14.1.8 RpcMethod.VTableIndex Property

Gets the V-table index of the method.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int VTableIndex { get; }
```

VB

```
Public ReadOnly Property VTableIndex As Integer
    Get
```

Property Value

Type: Int32
The index of the v table.

Reference

RpcMethod Class [▶ 1021]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.14.2 RpcMethod Methods

The RpcMethod [▶ 1021] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString [▶ 1028]	Returns a String that represents this instance. (Overrides Object.ToString..)

Reference

RpcMethod Class [▶ 1021]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.14.2.1 RpcMethod.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

RpcMethod Class [▶ 1021]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.15 RpcMethodParameter Class

Class RpcMethodParameter.

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.RpcMethodParameter
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class RpcMethodParameter : IRpcMethodParameter
```

VB

```
Public Class RpcMethodParameter
    Implements IRpcMethodParameter
```

The RpcMethodParameter type exposes the following members.

Properties

	Name	Description
	AlignSize [▶ 1030]	Gets the size of biggest element for alignment
	Comment [▶ 1031]	Gets the Parameter Comment.
	Name [▶ 1031]	Gets the Parameter Name
	ParameterFlags [▶ 1032]	Gets the parameter flags.
	Size [▶ 1032]	Gets the size of the RpcMethodParameter
	TypeGuid [▶ 1033]	Gets the Unique identifier of the parameters data type.
	TypeName [▶ 1033]	Gets the Data type of the Parameter

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.15.1 RpcMethodParameter Properties

The RpcMethodParameter [▶ 1028] type exposes the following members.

Properties

	Name	Description
	AlignSize [▶ 1030]	Gets the size of biggest element for alignment
	Comment [▶ 1031]	Gets the Parameter Comment.
	Name [▶ 1031]	Gets the Parameter Name
	ParameterFlags [▶ 1032]	Gets the parameter flags.
	Size [▶ 1032]	Gets the size of the RpcMethodParameter [▶ 1028]
	TypeGuid [▶ 1033]	Gets the Unique identifier of the parameters data type.
	TypeName [▶ 1033]	Gets the Data type of the Parameter

Reference

RpcMethodParameter Class [▶ 1028]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.15.1.1 RpcMethodParameter.AlignSize Property

Gets the size of biggest element for alignment

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int AlignSize { get; }
```

VB

```
Public ReadOnly Property AlignSize As Integer
    Get
```

Property Value

Type: Int32
The size of the align.

Reference

RpcMethodParameter Class [▶ 1028]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.15.1.2 RpcMethodParameter.Comment Property

Gets the Parameter Comment.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Comment { get; }
```

VB

```
Public ReadOnly Property Comment As String
    Get
```

Property Value

Type: String
The comment.

Reference

RpcMethodParameter Class [▶ 1028]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.15.1.3 RpcMethodParameter.Name Property

Gets the Parameter Name

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Name { get; }
```

VB

```
Public ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Implements

IRpcMethodParameter.Name [▶ 1602]

Reference

RpcMethodParameter Class [▶ 1028]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.15.1.4 RpcMethodParameter.ParameterFlags Property

Gets the parameter flags.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public MethodParamFlags ParameterFlags { get; }
```

VB

```
Public ReadOnly Property ParameterFlags As MethodParamFlags
    Get
```

Property Value

Type: MethodParamFlags [▶ 1724]
The parameter flags.

Implements

IRpcMethodParameter.ParameterFlags [▶ 1603]

Reference

RpcMethodParameter Class [▶ 1028]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.15.1.5 RpcMethodParameter.Size Property

Gets the size of the RpcMethodParameter [▶ 1028]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Size { get; }
```

VB

```
Public ReadOnly Property Size As Integer
    Get
```

Property Value

Type: Int32
The size.

Implements

IRpcMethodParameter.Size [▶ 1603]

Reference

RpcMethodParameter Class [▶ 1028]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.15.1.6 RpcMethodParameter.TypeGuid Property

Gets the Unique identifier of the parameters data type.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Guid TypeGuid { get; }
```

VB

```
Public ReadOnly Property TypeGuid As Guid
    Get
```

Property Value

Type: Guid
The type unique identifier.

Reference

RpcMethodParameter Class [▶ 1028]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.15.1.7 RpcMethodParameter.TypeName Property

Gets the Data type of the Parameter

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string TypeName { get; }
```

VB

```
Public ReadOnly Property TypeName As String
    Get
```

Property Value

Type: String
The type.

Implements

IRpcMethodParameter.TypeName [▶ 1604]

Reference

RpcMethodParameter Class [▶ 1028]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.15.2 RpcMethodParameter Methods

The RpcMethodParameter [▶ 1028] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

RpcMethodParameter Class [▶ 1028]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.16 RpcStructType Class

StructType which is callable by RPC Methods.

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.StructType [▶ 1044]
      TwinCAT.Ads.TypeSystem.RpcStructType
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class RpcStructType : StructType,
    IRpcCallableType
```

VB

```
Public NotInheritable Class RpcStructType
    Inherits StructType
    Implements IRpcCallableType
```

The RpcStructType type exposes the following members.

Properties

	Name	Description
	AllMembers [▶ 1047]	Gets all members (down the derivation hierarchy) (Inherited from StructType [▶ 1044].)
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BaseType [▶ 1048]	Gets the structs Base Type (Null if not derived). (Inherited from StructType [▶ 1044].)
	BaseTypeName [▶ 1048]	Gets the the Name of the Base class (if derived) (Inherited from StructType [▶ 1044].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	HasRpcMethods [▶ 1037]	Gets a value indicating whether this instance has RPC Methods. (Overrides StructType.HasRpcMethods [▶ 1049].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsDerived [▶ 1050]	Gets a value indicating whether this instance is derived. (Inherited from StructType [▶ 1044].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Members [▶ 1050]	Gets a read only collection of the Members [▶ 1562] of the IStructType [▶ 1619]. (Inherited from StructType [▶ 1044].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)

	Name	Description
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	RpcMethods [▶ 1038]	Gets the Method descriptions for the IRpcCallableType [▶ 1596]
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.16.1 RpcStructType Properties

The RpcStructType [▶ 1034] type exposes the following members.

Properties

	Name	Description
	AllMembers [▶ 1047]	Gets all members (down the derivation hierarchy) (Inherited from StructType [▶ 1044].)
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BaseType [▶ 1048]	Gets the structs Base Type (Null if not derived). (Inherited from StructType [▶ 1044].)
	BaseTypeName [▶ 1048]	Gets the the Name of the Base class (if derived) (Inherited from StructType [▶ 1044].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	HasRpcMethods [▶ 1037]	Gets a value indicating whether this instance has RPC Methods. (Overrides StructType.HasRpcMethods [▶ 1049].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)

	Name	Description
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsDerived [▶ 1050]	Gets a value indicating whether this instance is derived. (Inherited from StructType [▶ 1044].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Members [▶ 1050]	Gets a read only collection of the Members [▶ 1562] of the IStructType [▶ 1619]. (Inherited from StructType [▶ 1044].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	RpcMethods [▶ 1038]	Gets the Method descriptions for the IRpcCallableType [▶ 1596]
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Reference

RpcStructType Class [▶ 1034]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.16.1.1 RpcStructType.HasRpcMethods Property

Gets a value indicating whether this instance has RPC Methods.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool HasRpcMethods { get; }
```

VB

```
Public Overrides ReadOnly Property HasRpcMethods As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has methods; otherwise, false.

Implements

IStructType.HasRpcMethods [▶ 1623]

Remarks

The DataType (Structure) must be marked with the PlcAttribute 'TcRpcEnable' to enable RpcMethods, otherwise RpcMethods are not passed through to the ADS symbolic information.

Reference

RpcStructType Class [▶ 1034]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.16.1.2 RpcStructType.RpcMethods Property

Gets the Method descriptions for the IRpcCallableType [▶ 1596]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyRpcMethodCollection RpcMethods { get; }
```

VB

```
Public ReadOnly Property RpcMethods As ReadOnlyRpcMethodCollection
    Get
```

Property Value

Type: ReadOnlyRpcMethodCollection [▶ 1762]
The methods.

Implements

IRpcCallableType.RpcMethods [▶ 1597]

Remarks

The DataType (Structure) must be marked with the PlcAttribute 'TcRpcEnable' to enable RpcMethods, otherwise RpcMethods are not passed through to the ADS symbolic information.

Reference

RpcStructType Class [▶ 1034]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.16.2 RpcStructType Methods

The RpcStructType [▶ 1034] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

RpcStructType Class [▶ 1034]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.17 StringType Class

String DataType

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.StringType
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class StringType : DataType,
    IStringType, IDataType, IBitSize
```

VB

```
Public NotInheritable Class StringType
    Inherits DataType
    Implements IStringType, IDataType, IBitSize
```

The StringType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)

	Name	Description
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	Encoding [▶ 1042]	Gets the encoding of the String (Encoding.ASCII (STRING) or Encoding.UNICODE (WSTRING))
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsFixedLength [▶ 1042]	Gets a value indicating whether the string is of fixed length.
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	Length [▶ 1043]	Gets the number of characters within the string.
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 1044]	Returns a String that represents this instance. (Overrides DataType.ToString. [▶ 960].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.17.1 StringType Properties

The StringType [▶ 1039] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	Encoding [▶ 1042]	Gets the encoding of the String (Encoding.ASCII (STRING) or Encoding.UNICODE (WSTRING))
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsFixedLength [▶ 1042]	Gets a value indicating whether the string is of fixed length.
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	Length [▶ 1043]	Gets the number of characters within the string.
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Reference

StringType Class [▶ 1039]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.17.1.1 StringType.Encoding Property

Gets the encoding of the String (Encoding.ASCII (STRING) or Encoding.UNICODE (WSTRING))

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Encoding Encoding { get; }
```

VB

```
Public ReadOnly Property Encoding As Encoding
    Get
```

Property Value

Type: Encoding
The encoding.

Implements

IStringType.Encoding [▶ 1614]

Reference

StringType Class [▶ 1039]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.17.1.2 StringType.IsFixedLength Property

Gets a value indicating whether the string is of fixed length.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsFixedLength { get; }
```

VB

```
Public ReadOnly Property IsFixedLength As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is fixed length; otherwise, false.

Implements

IStringType.IsFixedLength [▶ 1615]

Exceptions

Exception	Condition
NotImplementedException	

Reference

StringType Class [▶ 1039]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.17.1.3 StringType.Length Property

Gets the number of characters within the string.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Length { get; }
```

VB

```
Public ReadOnly Property Length As Integer
    Get
```

Property Value

Type: Int32
The length.

Implements

IStringType.Length [▶ 1615]

Reference

StringType Class [▶ 1039]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.17.2 StringType Methods

The StringType [▶ 1039] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)

	Name	Description
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 1044]	Returns a String that represents this instance. (Overrides DataType.ToString. [▶ 960].)

Reference

StringType Class [▶ 1039]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.17.2.1 StringType.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

StringType Class [▶ 1039]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.18 StructType Class

Represents a struct type

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.StructType
      TwinCAT.Ads.TypeSystem.RpcStructType [▶ 1034]
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class StructType : DataType, IStructType,
    IDataType, IBitSize
```

VB

```
Public Class StructType
    Inherits DataType
    Implements IStructType, IDataType, IBitSize
```

The StructType type exposes the following members.

Properties

	Name	Description
	AllMembers [▶ 1047]	Gets all members (down the derivation hierarchy)
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BaseType [▶ 1048]	Gets the structs Base Type (Null if not derived).
	BaseTypeName [▶ 1048]	Gets the the Name of the Base class (if derived)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	HasRpcMethods [▶ 1049]	Gets a value indicating whether this StructType has RPC Methods.
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsDerived [▶ 1050]	Gets a value indicating whether this instance is derived.
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Members [▶ 1050]	Gets a read only collection of the Members [▶ 1562] of the IStructType [▶ 1619].
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)

	Name	Description
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.18.1 StructType Properties

The StructType [▶ 1044] type exposes the following members.

Properties

	Name	Description
	AllMembers [▶ 1047]	Gets all members (down the derivation hierarchy)
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BaseType [▶ 1048]	Gets the structs Base Type (Null if not derived).
	BaseTypeName [▶ 1048]	Gets the the Name of the Base class (if derived)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	HasRpcMethods [▶ 1049]	Gets a value indicating whether this StructType [▶ 1044] has RPC Methods.

	Name	Description
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsDerived [▶ 1050]	Gets a value indicating whether this instance is derived.
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Members [▶ 1050]	Gets a read only collection of the Members [▶ 1562] of the IStructType [▶ 1619].
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Reference

StructType Class [▶ 1044]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.18.1.1 StructType.AllMembers Property

Gets all members (down the derivation hierarchy)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyMemberCollection AllMembers { get; }
```

VB

```
Public ReadOnly Property AllMembers As ReadOnlyMemberCollection
    Get
```

Property Value

Type: ReadOnlyMemberCollection [▶ 1755]
All members.

Implements

IStructType.AllMembers [▶ 1622]

Reference

StructType Class [▶ 1044]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.18.1.2 StructType.BaseType Property

Gets the structs Base Type (Null if not derived).

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType BaseType { get; }
```

VB

```
Public ReadOnly Property BaseType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]

Implements

IStructType.BaseType [▶ 1622]

Reference

StructType Class [▶ 1044]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.18.1.3 StructType.BaseTypeName Property

Gets the the Name of the Base class (if derived)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string BaseTypeName { get; }
```

VB

```
Public ReadOnly Property BaseTypeName As String
    Get
```

Property Value

Type: String
Empty if not derived.

Implements

IStructType.BaseTypeName [▶ 1623]

Reference

StructType Class [▶ 1044]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.18.1.4 StructType.HasRpcMethods Property

Gets a value indicating whether this StructType [▶ 1044] has RPC Methods.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual bool HasRpcMethods { get; }
```

VB

```
Public Overridable ReadOnly Property HasRpcMethods As Boolean
    Get
```

Property Value

Type: Boolean
true if this type has methods; otherwise, false.

Implements

IStructType.HasRpcMethods [▶ 1623]

Remarks

The DataType (Structure) must be marked with the PlcAttribute 'TcRpcEnable' to enable RpcMethods, otherwise RpcMethods are not passed through to the ADS symbolic information.

Reference

StructType Class [▶ 1044]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.18.1.5 StructType.IsDerived Property

Gets a value indicating whether this instance is derived.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsDerived { get; }
```

VB

```
Public ReadOnly Property IsDerived As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is derived; otherwise, false.

Reference

StructType Class [▶ 1044]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.18.1.6 StructType.Members Property

Gets a read only collection of the Members [▶ 1562] of the IStructType [▶ 1619].

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyMemberCollection Members { get; }
```

VB

```
Public ReadOnly Property Members As ReadOnlyMemberCollection
    Get
```

Property Value

Type: ReadOnlyMemberCollection [▶ 1755]
The members as read only collection.

Implements

IStructType.Members [▶ 1624]

Remarks

If the IStructType [▶ 1619] is derived, only the extended members are returned. To get all supported members down the inheritance chain, use the AllMembers [▶ 1622] property.

Reference

StructType Class [▶ 1044]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.18.2 StructType Methods

The StructType [▶ 1044] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

StructType Class [▶ 1044]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.19 SubRangeType.T. Class

Represents a SubRangType

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.SubRangeType.T.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class SubRangeType<T> : DataType,
    ISubRangeType<T>, ISubRangeType, IDataType, IBitSize
where T : struct, new()
```

VB

```
Public NotInheritable Class SubRangeType(Of T As {Structure, New})
    Inherits DataType
    Implements ISubRangeType(Of T), ISubRangeType, IDataType,
    IBitSize
```

Type Parameters

T

The SubRangeType.T. type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BaseType [▶ 1054]	Gets the the base type.
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	LowerBound [▶ 1055]	Gets the lower bound.
	ManagedType [▶ 1055]	Gets the corresponding .NET Type if attached. (Overrides DataType.ManagedType [▶ 956].)

	Name	Description
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)
	UpperBound [▶ 1056]	Gets the upper bound.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.19.1 SubRangeType.T. Properties

The SubRangeType.T. [▶ 1051] generic type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BaseType [▶ 1054]	Gets the the base type.
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)

	Name	Description
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	LowerBound [▶ 1055]	Gets the lower bound.
	ManagedType [▶ 1055]	Gets the corresponding .NET Type if attached. (Overrides DataType.ManagedType [▶ 956].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)
	UpperBound [▶ 1056]	Gets the upper bound.

Reference

SubRangeType.T. Class [▶ 1051]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.19.1.1 SubRangeType.T..BaseType Property

Gets the the base type.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType BaseType { get; }
```

VB

```
Public ReadOnly Property BaseType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the referenced.

Implements

ISubRangeType.BaseType [▶ 1630]

Reference

SubRangeType.T. Class [▶ 1051]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.19.1.2 SubRangeType.T..LowerBound Property

Gets the lower bound.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T LowerBound { get; }
```

VB

```
Public ReadOnly Property LowerBound As T
    Get
```

Property Value

Type: T [▶ 1051]
The lower bound.

Implements

ISubRangeType.T..LowerBound [▶ 1633]

Reference

SubRangeType.T. Class [▶ 1051]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.19.1.3 SubRangeType.T..ManagedType Property

Gets the corresponding .NET Type if attached.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override Type ManagedType { get; }
```

VB

```
Public Overrides ReadOnly Property ManagedType As Type
    Get
```

Property Value

Type: Type
Dot net type.

Reference

SubRangeType.T. Class [▶ 1051]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.19.1.4 SubRangeType.T..UpperBound Property

Gets the upper bound.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T UpperBound { get; }
```

VB

```
Public ReadOnly Property UpperBound As T
    Get
```

Property Value

Type: T [▶ 1051]
The lower bound.

Implements

ISubRangeType.T..UpperBound [▶ 1633]

Reference

SubRangeType.T. Class [▶ 1051]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.19.2 SubRangeType.T. Methods

The SubRangeType.T. [▶ 1051] generic type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)

	Name	Description
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

SubRangeType.T. Class [▶ 1051]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20 Symbol Class

Symbol class

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.Instance [▶ 982]
    TwinCAT.Ads.TypeSystem.Symbol
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class Symbol : Instance, IValueSymbol3,
    IValueSymbol2, IValueSymbol, IValueRawSymbol, ISymbol, IAttributedInstance,
    IInstance, IBitSize, IValueAnySymbol, IAdsSymbol, IProcessImageAddress
```

VB

```
Public Class Symbol
    Inherits Instance
    Implements IValueSymbol3, IValueSymbol2, IValueSymbol, IValueRawSymbol,
    ISymbol, IAttributedInstance, IInstance, IBitSize, IValueAnySymbol,
    IAdsSymbol, IProcessImageAddress
```

The Symbol type exposes the following members.

Properties

	Name	Description
	AccessRights [▶ 1064]	Gets the access rights.
	Attributes [▶ 985]	Gets the Type Attributes. (Inherited from Instance [▶ 982].)
	BitSize [▶ 986]	Gets the size of this Instance [▶ 982] in bits. (Inherited from Instance [▶ 982].)
	ByteSize [▶ 986]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from Instance [▶ 982].)
	Category [▶ 987]	Gets the the DataTypeCategory [▶ 1189] of the Instance. (Inherited from Instance [▶ 982].)
	Comment [▶ 987]	Gets the comment. (Inherited from Instance [▶ 982].)
	Connection [▶ 1064]	Gets the connection that produces values for this IValueSymbol [▶ 1683]
	ContextMask [▶ 988]	Gets the context mask of this instance. (Inherited from Instance [▶ 982].)

	Name	Description
	DataType [▶ 989]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from Instance [▶ 982].)
	HasValue [▶ 989]	Gets a value indicating whether this instance has a value. (Inherited from Instance [▶ 982].)
	ImageBaseAddress [▶ 1065]	Gets the AmsAddress [▶ 410] of the Process Image
	IndexGroup [▶ 1065]	Gets the index group of the Symbol
	IndexOffset [▶ 1066]	Gets the index offset of the Symbol
	InstanceName [▶ 990]	Gets the name of the instance (without periods (.) (Inherited from Instance [▶ 982].)
	InstancePath [▶ 1066]	Gets the relative / absolute access path to the instance (with periods (.)) (Overrides Instance.InstancePath [▶ 990].)
	IsBitType [▶ 991]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from Instance [▶ 982].)
	IsByteAligned [▶ 991]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from Instance [▶ 982].)
	IsContainerType [▶ 1067]	Gets a value indicating whether the Symbols datatype is a Container type.
	IsDereferencedPointer [▶ 1068]	Gets or sets a value indicating whether an ancestor is a dereferenced Pointer
	IsDereferencedReference [▶ 1068]	Gets or sets a value indicating whether an ancestor is a dereferenced Reference
	IsPersistent [▶ 992]	Indicates if this instance is persistent. (Inherited from Instance [▶ 982].)
	IsPointer [▶ 992]	Gets a value indicating whether this instance is reference. (Inherited from Instance [▶ 982].)
	IsPrimitiveType [▶ 1069]	Gets a value indicating whether this instance is primitive.
	IsReadOnly [▶ 993]	Indicates if this instance is read only. (Inherited from Instance [▶ 982].)
	IsRecursive [▶ 1069]	Gets a value indicating whether this instance is recursive.
	IsReference [▶ 993]	Gets a value indicating whether this instance is reference. (Inherited from Instance [▶ 982].)
	IsStatic [▶ 994]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from Instance [▶ 982].)
	IsTcComInterfacePointer [▶ 995]	Indicates if this instance is a TcComInterfacePointer. (Inherited from Instance [▶ 982].)
	IsTypeGuid [▶ 995]	Indicates if this instance has set TypeGuid flag. (Inherited from Instance [▶ 982].)
	Namespace [▶ 996]	Gets the namespace name. (Inherited from Instance [▶ 982].)
	NotificationSettings [▶ 1070]	Gets or sets the notification settings.

	Name	Description
	Parent [▶ 1071]	Gets the parent Symbol
	Size [▶ 996]	Gets the size of the IDataType [▶ 1517] in bytes or Bits dependant on IsBitType [▶ 991] (Inherited from Instance [▶ 982].)
	SubSymbolCount [▶ 1071]	Gets the number of SubSymbols
	SubSymbols [▶ 1072]	Gets the SubSymbols of the ISymbol [▶ 1634]
	TypeName [▶ 997]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from Instance [▶ 982].)

Methods

	Name	Description
	EnsureRights [▶ 1075]	Ensures that the AccessRights are matched.
	Equals [▶ 1076]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 1076]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnGetSize [▶ 998]	Handler function getting the size of the Instance [▶ 982] (Inherited from Instance [▶ 982].)
	OnReadRawValue [▶ 1077]	Handler function for reading the raw value
	OnReadValue [▶ 1077]	Handler function for reading the dynamic value.
	OnSetInstanceName [▶ 1078]	Sets a new InstanceName InstancePath (Overrides Instance.OnSetInstanceName(String) [▶ 999].)
	OnTryReadValue [▶ 1078]	Handler function for reading the dynamic value.
	OnTryWriteValue [▶ 1079]	Handler function for writing the dynamic value
	OnWriteRawValue [▶ 1080]	Handler function for writing the RawValue
	OnWriteValue [▶ 1081]	Handler function for writing the dynamic value
	ReadAnyValue(Type) [▶ 1082]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type
	ReadAnyValue(Type, Int32) [▶ 1083]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type
	ReadRawValue. [▶ 1084]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

	Name	Description
	ReadRawValue(Int32) [▶ 1085]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	ReadValue. [▶ 1086]	Reads the Value of the IValueSymbol [▶ 1683]
	ReadValue(Int32) [▶ 1086]	Reads the Value of the IValueSymbol [▶ 1683]
	SetContextMask [▶ 999]	Sets the context mask. (Inherited from Instance [▶ 982].)
	SetParent [▶ 1087]	Sets the parent symbol.
	ToString [▶ 1088]	Returns a String that represents this instance. (Overrides Instance.ToString. [▶ 1000].)
	TryReadValue [▶ 1088]	Reads the Value of the IValueSymbol [▶ 1683]
	TryWriteValue [▶ 1089]	Writes the Value of the IValueSymbol [▶ 1683]
	UpdateAnyValue(Object.) [▶ 1091]	Reads the value of this Value [▶ 1683] into the specified managed value.
	UpdateAnyValue(Object., Int32) [▶ 1091]	Reads the value of this Value [▶ 1683] into the specified managed value.
	WriteRawValue(.Byte.) [▶ 1092]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	WriteRawValue(.Byte., Int32) [▶ 1093]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	WriteValue(Object) [▶ 1094]	Writes the Value of the IValueSymbol [▶ 1683]
	WriteValue(Object, Int32) [▶ 1095]	Writes the Value of the IValueSymbol [▶ 1683]

Events

	Name	Description
	RawValueChanged [▶ 1096]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed.
	ValueChanged [▶ 1097]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed.

Operators

	Name	Description
	Equality [▶ 1098]	Operator==
	Inequality [▶ 1099]	Implements the != operator.

Fields

	Name	Description
	attributes [▶ 1001]	The attributes (Inherited from Instance [▶ 982].)

Extension Methods

	Name	Description
	PollValuesAnnotated(IObservable.Unit.) [▶ 882]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence (Defined by ValueSymbolExtensions [▶ 877].)
	PollValuesAnnotated(TimeSpan) [▶ 883]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time. (Defined by ValueSymbolExtensions [▶ 877].)
	WhenValueChanged [▶ 884]	Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object.) [▶ 888]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception.) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., CancellationToken) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception., CancellationToken) [▶ 891]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)

Remarks

A Symbol is a (named) memory object within the Process Image with a fixed address indicated by Index Group and Index Offset. Symbols can optionally be addressed by instance path and are bound to a specific DataType [▶ 945].

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

TwinCAT.Ads.TypeSystem.Instance [▶ 982]

TwinCAT.TypeSystem.IValueSymbol [▶ 1683]

TwinCAT.TypeSystem.IValueAnySymbol [▶ 1666]

IValueAccessorProvider

ISymbolFactoryServicesProvider

IHierarchicalSymbol

ISymbolValueChangeNotify

IContextMaskProvider

IInstanceInternal

ISymbolInternal

TwinCAT.Ads.TypeSystem.IAdsSymbol [▶ 976]

5.5.20.1 Symbol Properties

The Symbol [▶ 1057] type exposes the following members.

Properties

	Name	Description
	AccessRights [▶ 1064]	Gets the access rights.
	Attributes [▶ 985]	Gets the Type Attributes. (Inherited from Instance [▶ 982].)
	BitSize [▶ 986]	Gets the size of this Instance [▶ 982] in bits. (Inherited from Instance [▶ 982].)
	ByteSize [▶ 986]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from Instance [▶ 982].)
	Category [▶ 987]	Gets the the DataTypeCategory [▶ 1189] of the Instance. (Inherited from Instance [▶ 982].)
	Comment [▶ 987]	Gets the comment. (Inherited from Instance [▶ 982].)
	Connection [▶ 1064]	Gets the connection that produces values for this IValueSymbol [▶ 1683]
	ContextMask [▶ 988]	Gets the context mask of this instance. (Inherited from Instance [▶ 982].)
	DataType [▶ 989]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from Instance [▶ 982].)
	HasValue [▶ 989]	Gets a value indicating whether this instance has a value. (Inherited from Instance [▶ 982].)
	ImageBaseAddress [▶ 1065]	Gets the AmsAddress [▶ 410] of the Process Image
	IndexGroup [▶ 1065]	Gets the index group of the Symbol
	IndexOffset [▶ 1066]	Gets the index offset of the Symbol
	InstanceName [▶ 990]	Gets the name of the instance (without periods (.) (Inherited from Instance [▶ 982].)
	InstancePath [▶ 1066]	Gets the relative / absolute access path to the instance (with periods (.)) (Overrides Instance.InstancePath [▶ 990].)
	IsBitType [▶ 991]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from Instance [▶ 982].)

	Name	Description
	IsByteAligned [▶ 991]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from Instance [▶ 982].)
	IsContainerType [▶ 1067]	Gets a value indicating whether the Symbols datatype is a Container type.
	IsDereferencedPointer [▶ 1068]	Gets or sets a value indicating whether an ancestor is a dereferenced Pointer
	IsDereferencedReference [▶ 1068]	Gets or sets a value indicating whether an ancestor is a dereferenced Reference
	IsPersistent [▶ 992]	Indicates if this instance is persistent. (Inherited from Instance [▶ 982].)
	IsPointer [▶ 992]	Gets a value indicating whether this instance is reference. (Inherited from Instance [▶ 982].)
	IsPrimitiveType [▶ 1069]	Gets a value indicating whether this instance is primitive.
	IsReadOnly [▶ 993]	Indicates if this instance is read only. (Inherited from Instance [▶ 982].)
	IsRecursive [▶ 1069]	Gets a value indicating whether this instance is recursive.
	IsReference [▶ 993]	Gets a value indicating whether this instance is reference. (Inherited from Instance [▶ 982].)
	IsStatic [▶ 994]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from Instance [▶ 982].)
	IsTcComInterfacePointer [▶ 995]	Indicates if this instance is a TcComInterfacePointer. (Inherited from Instance [▶ 982].)
	IsTypeGuid [▶ 995]	Indicates if this instance has set TypeGuid flag. (Inherited from Instance [▶ 982].)
	Namespace [▶ 996]	Gets the namespace name. (Inherited from Instance [▶ 982].)
	NotificationSettings [▶ 1070]	Gets or sets the notification settings.
	Parent [▶ 1071]	Gets the parent Symbol
	Size [▶ 996]	Gets the size of the IDataType [▶ 1517] in bytes or Bits dependant on IsBitType [▶ 991] (Inherited from Instance [▶ 982].)
	SubSymbolCount [▶ 1071]	Gets the number of SubSymbols
	SubSymbols [▶ 1072]	Gets the SubSymbols of the ISymbol [▶ 1634]
	TypeName [▶ 997]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from Instance [▶ 982].)

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.1 Symbol.AccessRights Property

Gets the access rights.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolAccessRights AccessRights { get; }
```

VB

```
Public ReadOnly Property AccessRights As SymbolAccessRights
    Get
```

Property Value

Type: SymbolAccessRights [▶ 1818]
The access rights.

Implements

IValueSymbol.AccessRights [▶ 1687]

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.2 Symbol.Connection Property

Gets the connection that produces values for this IValueSymbol [▶ 1683]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IConnection Connection { get; }
```

VB

```
Public ReadOnly Property Connection As IConnection
    Get
```

Property Value

Type: IConnection [▶ 46]
The connection object.

Implements

IValueSymbol2.Connection [▶ 1698]

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.3 Symbol.ImageBaseAddress Property

Gets the AmsAddress [▶ 410] of the Process Image

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public AmsAddress ImageBaseAddress { get; }
```

VB

```
Public ReadOnly Property ImageBaseAddress As AmsAddress
    Get
```

Property Value

Type: AmsAddress [▶ 410]
The address.

Implements

IAdsSymbol.ImageBaseAddress [▶ 979]

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.4 Symbol.IndexGroup Property

Gets the index group of the Symbol

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public uint IndexGroup { get; }
```

VB

```
Public ReadOnly Property IndexGroup As UInteger
    Get
```

Property Value

Type: UInt32
The index group.

Implements

IProcessImageAddress.IndexGroup [▶ 1581]

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.5 Symbol.IndexOffset Property

Gets the index offset of the Symbol

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public uint IndexOffset { get; }
```

VB

```
Public ReadOnly Property IndexOffset As UInteger
    Get
```

Property Value

Type: UInt32
The index offset.

Implements

IProcessImageAddress.IndexOffset [▶ 1581]

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.6 Symbol.InstancePath Property

Gets the relative / absolute access path to the instance (with periods (.))

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string InstancePath { get; }
```

VB

```
Public Overrides ReadOnly Property InstancePath As String
    Get
```

Property Value

Type: String
The instance path.

Implements

IInstance.InstancePath [▶ 1560]
IInstance.InstancePath [▶ 1560]

Remarks

If this path is relative or absolute depends on the context. IMember [▶ 1562] are using relative paths, ISymbol [▶ 1634]s are using absolute ones.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.7 Symbol.IsContainerType Property

Gets a value indicating whether the Symbols datatype is a Container type.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual bool IsContainerType { get; }
```

VB

```
Public Overridable ReadOnly Property IsContainerType As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is container type; otherwise, false.

Implements

ISymbol.IsContainerType [▶ 1637]

Remarks

Container Types are all types that contain SubElements like

- Array [▶ 1189]
- Pointer [▶ 1189]
- Union [▶ 1189]
- Struct [▶ 1189]
- Function [▶ 1189]
- FunctionBlock [▶ 1189]
- Program [▶ 1189]

and the Alias [▶ 1189] and Reference [▶ 1189] types, if they have a container type as base type.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

IDataType.Category [▶ 1520]

5.5.20.1.8 Symbol.IsDereferencedPointer Property

Gets or sets a value indicating whether an ancestor is a dereferenced Pointer

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsDereferencedPointer { get; }
```

VB

```
Public ReadOnly Property IsDereferencedPointer As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is ancestor is pointer; otherwise, false.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.9 Symbol.IsDereferencedReference Property

Gets or sets a value indicating whether an ancestor is a dereferenced Reference

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsDereferencedReference { get; }
```

VB

```
Public ReadOnly Property IsDereferencedReference As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is ancestor is reference; otherwise, false.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.10 Symbol.IsPrimitiveType Property

Gets a value indicating whether this instance is primitive.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual bool IsPrimitiveType { get; }
```

VB

```
Public Overridable ReadOnly Property IsPrimitiveType As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is primitive; otherwise, false.

Implements

ISymbol.IsPrimitiveType [▶ 1638]

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.11 Symbol.IsRecursive Property

Gets a value indicating whether this instance is recursive.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsRecursive { get; }
```

VB

```
Public ReadOnly Property IsRecursive As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is recursive; otherwise, false.

Implements

ISymbol.IsRecursive [▶ 1639]

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.12 Symbol.NotificationSettings Property

Gets or sets the notification settings.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public INotificationSettings NotificationSettings { get; set; }
```

VB

```
Public Property NotificationSettings As INotificationSettings
    Get
    Set
```

Property Value

Type: INotificationSettings [▶ 1566]
The notification settings.

Implements

IValueSymbol.NotificationSettings [▶ 1688]

Remarks

The NotificationSettings will be inherited from Parent [▶ 1071] if the setting is not overwritten. If the Root Symbol also doesn't contain the settings, then the DefaultNotificationSettings [▶ 981] will be returned.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.13 Symbol.Parent Property

Gets the parent Symbol

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ISymbol Parent { get; }
```

VB

```
Public ReadOnly Property Parent As ISymbol
    Get
```

Property Value

Type: ISymbol [▶ 1634]
The parent.

Implements

ISymbol.Parent [▶ 1639]

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.14 Symbol.SubSymbolCount Property

Gets the number of SubSymbols

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int SubSymbolCount { get; }
```

VB

```
Public ReadOnly Property SubSymbolCount As Integer
    Get
```

Property Value

Type: Int32
The Number of SubSymbols.

Remarks

If the SubSymbols [▶ 1072] collection is not generated yet (WeakReference), then this method is less memory and cpu consuming to use for just determining the the number of child symbols (instead of using SubSymbols.Count)/>

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.1.15 Symbol.SubSymbols Property

Gets the SubSymbols of the ISymbol [▶ 1634]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlySymbolCollection SubSymbols { get; }
```

VB

```
Public ReadOnly Property SubSymbols As ReadOnlySymbolCollection
    Get
```

Property Value

Type: ReadOnlySymbolCollection [▶ 1772]

Implements

ISymbol.SubSymbols [▶ 1640]

Remarks

Used for Array, Struct, Pointer and Reference instances. Otherwise empty

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2 Symbol Methods

The Symbol [▶ 1057] type exposes the following members.

Methods

	Name	Description
	EnsureRights [▶ 1075]	Ensures that the AccessRights are matched.
	Equals [▶ 1076]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 1076]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnGetSize [▶ 998]	Handler function getting the size of the Instance [▶ 982] (Inherited from Instance [▶ 982].)
	OnReadRawValue [▶ 1077]	Handler function for reading the raw value
	OnReadValue [▶ 1077]	Handler function for reading the dynamic value.
	OnSetInstanceName [▶ 1078]	Sets a new InstanceName InstancePath (Overrides Instance.OnSetInstanceName(String) [▶ 999].)
	OnTryReadValue [▶ 1078]	Handler function for reading the dynamic value.
	OnTryWriteValue [▶ 1079]	Handler function for writing the dynamic value
	OnWriteRawValue [▶ 1080]	Handler function for writing the RawValue
	OnWriteValue [▶ 1081]	Handler function for writing the dynamic value
	ReadAnyValue(Type) [▶ 1082]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type
	ReadAnyValue(Type, Int32) [▶ 1083]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type
	ReadRawValue. [▶ 1084]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	ReadRawValue(Int32) [▶ 1085]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	ReadValue. [▶ 1086]	Reads the Value of the IValueSymbol [▶ 1683]
	ReadValue(Int32) [▶ 1086]	Reads the Value of the IValueSymbol [▶ 1683]
	SetContextMask [▶ 999]	Sets the context mask. (Inherited from Instance [▶ 982].)

	Name	Description
	SetParent [▶ 1087]	Sets the parent symbol.
	ToString [▶ 1088]	Returns a String that represents this instance. (Overrides Instance.ToString. [▶ 1000].)
	TryReadValue [▶ 1088]	Reads the Value of the IValueSymbol [▶ 1683]
	TryWriteValue [▶ 1089]	Writes the Value of the IValueSymbol [▶ 1683]
	UpdateAnyValue(Object.) [▶ 1091]	Reads the value of this Value [▶ 1683] into the specified managed value.
	UpdateAnyValue(Object., Int32) [▶ 1091]	Reads the value of this Value [▶ 1683] into the specified managed value.
	WriteRawValue(.Byte.) [▶ 1092]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	WriteRawValue(.Byte., Int32) [▶ 1093]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	WriteValue(Object) [▶ 1094]	Writes the Value of the IValueSymbol [▶ 1683]
	WriteValue(Object, Int32) [▶ 1095]	Writes the Value of the IValueSymbol [▶ 1683]

Extension Methods

	Name	Description
	PollValuesAnnotated(IObservable.Unit.) [▶ 882]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence (Defined by ValueSymbolExtensions [▶ 877].)
	PollValuesAnnotated(TimeSpan) [▶ 883]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time. (Defined by ValueSymbolExtensions [▶ 877].)
	WhenValueChanged [▶ 884]	Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object.) [▶ 888]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception.) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)

	Name	Description
	WriteValues(IObservable.Object., CancellationToken) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception., CancellationToken) [▶ 891]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.1 Symbol.EnsureRights Method

Ensures that the AccessRights are matched.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected void EnsureRights(
    SymbolAccessRights requested
)
```

VB

```
Protected Sub EnsureRights (
    requested As SymbolAccessRights
)
```

Parameters

requested	Type: TwinCAT.TypeSystem.SymbolAccessRights [▶ 1818] The requested rights.

Exceptions

Exception	Condition
InsufficientAccessRights [▶ 56]	

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.2 Symbol.Equals Method

Equals

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool Equals(
    Object obj
)
```

VB

```
Public Overrides Function Equals (
    obj As Object
) As Boolean
```

Parameters

obj	Type: System.Object The object to compare with the current object.

Return Value

Type: Boolean
true if the specified Object is equal to this instance; otherwise, false.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.3 Symbol.GetHashCode Method

Gets the HashCode of the Address

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override int GetHashCode()
```

VB

```
Public Overrides Function GetHashCode As Integer
```

Return Value

Type: Int32
A hash code for this instance, suitable for use in hashing algorithms and data structures like a hash table.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.4 Symbol.OnReadRawValue Method

Handler function for reading the raw value

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual byte[] OnReadRawValue(
    int timeout
)
```

VB

```
Protected Overridable Function OnReadRawValue (
    timeout As Integer
) As Byte()
```

Parameters

timeout	Type: System.Int32

Return Value

Type: .Byte.
System.Byte[].

Exceptions

Exception	Condition
AdsErrorException [▶ 310]	
AdsErrorException [▶ 310]	

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.5 Symbol.OnReadValue Method

Handler function for reading the dynamic value.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual Object OnReadValue(
    int timeout
)
```

VB

```
Protected Overridable Function OnReadValue (
    timeout As Integer
) As Object
```

Parameters

timeout	Type: System.Int32

Return Value

Type: Object
The Value

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.6 Symbol.OnSetInstanceName Method

Sets a new InstanceName InstancePath

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected override void OnSetInstanceName(
    string instanceName
)
```

VB

```
Protected Overrides Sub OnSetInstanceName (
    instanceName As String
)
```

Parameters

instanceName	Type: System.String Instance name.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.7 Symbol.OnTryReadValue Method

Handler function for reading the dynamic value.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual int OnTryReadValue(
    int timeout,
    out Object value
)
```

VB

```
Protected Overridable Function OnTryReadValue (
    timeout As Integer,
    <OutAttribute> ByRef value As Object
) As Integer
```

Parameters

timeout	Type: System.Int32
value	Type: System.Object.

Return Value

Type: Int32
The Value

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.8 Symbol.OnTryWriteValue Method

Handler function for writing the dynamic value

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual int OnTryWriteValue(
    Object value,
    int timeout
)
```

VB

```
Protected Overridable Function OnTryWriteValue (
    value As Object,
    timeout As Integer
) As Integer
```

Parameters

value	Type: System.Object The value.
timeout	Type: System.Int32 The timeout.

Return Value

Type: Int32

Exceptions

Exception	Condition
CannotAccessVirtualSymbolException [▶ 31]	
AdsSymbolException [▶ 389]	

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.9 Symbol.OnWriteRawValue Method

Handler function for writing the RawValue

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual void OnWriteRawValue(
    byte[] value,
    int timeout
)
```

VB

```
Protected Overridable Sub OnWriteRawValue (
    value As Byte(),
    timeout As Integer
)
```

Parameters

value	Type: .System.Byte. The value.
timeout	Type: System.Int32 The timeout.

Exceptions

Exception	Condition
CannotAccessVirtualSymbolException [▶ 31]	
SymbolException [▶ 606]	
SymbolException [▶ 606]	
AdsErrorException [▶ 310]	

Exception	Condition
AdsSymbolException [▶ 389]	
AdsErrorException [▶ 310]	

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.10 Symbol.OnWriteValue Method

Handler function for writing the dynamic value

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual void OnWriteValue(
    Object value,
    int timeout
)
```

VB

```
Protected Overridable Sub OnWriteValue (
    value As Object,
    timeout As Integer
)
```

Parameters

value	Type: System.Object The value.
timeout	Type: System.Int32 The timeout.

Exceptions

Exception	Condition
CannotAccessVirtualSymbolException [▶ 31]	
AdsSymbolException [▶ 389]	

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.11 Symbol.ReadAnyValue Method

Overload List

	Name	Description
	ReadAnyValue(Type) [▶ 1082]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type
	ReadAnyValue(Type, Int32) [▶ 1083]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

Symbol.ReadAnyValue Method (Type)

Reads the value of this Value [▶ 1683] into a new created instance of the managed type

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadAnyValue(
    Type managedType
)
```

VB

```
Public Function ReadAnyValue (
    managedType As Type
) As Object
```

Parameters

managedType	Type: System.Type The tp.

Return Value

Type: Object
Read value (System.Object).

Implements

IValueAnySymbol.ReadAnyValue(Type) [▶ 1670]

Reference

Symbol Class [▶ 1057]

ReadAnyValue Overload [▶ 1082]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

IValueAnySymbol.WriteAnyValue(Object) [▶ 1673]

IValueAnySymbol.UpdateAnyValue(Object.) [▶ 1672]

Symbol.ReadAnyValue Method (Type, Int32)

Reads the value of this Value [▶ 1683] into a new created instance of the managed type

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadAnyValue(
    Type managedType,
    int timeout
)
```

VB

```
Public Function ReadAnyValue (
    managedType As Type,
    timeout As Integer
) As Object
```

Parameters

managedType	Type: System.Type The tp.
timeout	Type: System.Int32 The timeout in ms.

Return Value

Type: Object
Read value (System.Object).

Implements

IValueAnySymbol.ReadAnyValue(Type, Int32) [▶ 1670]

Reference

Symbol Class [▶ 1057]

ReadAnyValue Overload [▶ 1082]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

IValueAnySymbol.WriteAnyValue(Object) [▶ 1673]

IValueAnySymbol.UpdateAnyValue(Object.) [▶ 1672]

5.5.20.2.12 Symbol.ReadRawValue Method

Overload List

	Name	Description
	ReadRawValue. [▶ 1084]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	ReadRawValue(Int32) [▶ 1085]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

Symbol.ReadRawValue Method

Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public byte[] ReadRawValue()
```

VB

```
Public Function ReadRawValue As Byte()
```

Field Value

Type: .Byte.
The raw value.

Return Value

Type: .Byte.
System.Byte[].

Implements

IValueRawSymbol.ReadRawValue. [▶ 1679]

Reference

Symbol Class [▶ 1057]

ReadRawValue Overload [▶ 1084]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

Symbol.ReadRawValue Method (Int32)

Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public byte[] ReadRawValue(
    int timeout
)
```

VB

```
Public Function ReadRawValue (
    timeout As Integer
) As Byte()
```

Parameters

timeout	Type: System.Int32

Field Value

Type: .Byte.
The raw value.

Return Value

Type: .Byte.
System.Byte[].

Implements

IValueRawSymbol.ReadRawValue(Int32) [▶ 1680]

Reference

Symbol Class [▶ 1057]

ReadRawValue Overload [▶ 1084]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.13 Symbol.ReadValue Method

Overload List

	Name	Description
	ReadValue. [▶ 1086]	Reads the Value of the IValueSymbol [▶ 1683]
	ReadValue(Int32) [▶ 1086]	Reads the Value of the IValueSymbol [▶ 1683]

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

Symbol.ReadValue Method

Reads the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadValue()
```

VB

```
Public Function ReadValue As Object
```

Field Value

Type: Object
The value.

Return Value

Type: Object
System.Object.

Implements

IValueSymbol.ReadValue. [▶ 1690]

Remarks

Calling on primitive types, a call of this method will return the primitive value. On complex types (structures and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: the raw byte Array will be returned, in dynamic mode: A Value will be created on the fly.

Reference

Symbol Class [▶ 1057]

ReadValue Overload [▶ 1085]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

Symbol.ReadValue Method (Int32)

Reads the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadValue(
    int timeout
)
```

VB

```
Public Function ReadValue (
    timeout As Integer
) As Object
```

Parameters

timeout	Type: System.Int32 The timeout in ms.

Field Value

Type: Object
The value.

Return Value

Type: Object
System.Object.

Implements

IValueSymbol.ReadValue(Int32) [▶ 1691]

Remarks

Calling on primitive types, a call of this method will return the primitive value. On complex types (structures and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: the raw byte Array will be returned, in dynamic mode: A Value will be created on the fly. A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

Symbol Class [▶ 1057]

ReadValue Overload [▶ 1085]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.14 Symbol.SetParent Method

Sets the parent symbol.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void SetParent(
    ISymbol parent
)
```

VB

```
Public Sub SetParent (
    parent As ISymbol
)
```

Parameters

parent	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The parent.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.15 Symbol.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.16 Symbol.TryReadValue Method

Reads the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int TryReadValue(
    int timeout,
    out Object value
)
```

VB

```
Public Function TryReadValue (
    timeout As Integer,
    <OutAttribute> ByRef value As Object
) As Integer
```

Parameters

timeout	Type: System.Int32 The timeout in ms.
value	Type: System.Object. The read value.

Return Value

Type: Int32
The error Code..

Implements

IValueSymbol3.TryReadValue(Int32, Object.) [▶ 1706]

Remarks

Calling on primitive types, a call of this method will return the primitive value. On complex types (structures and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: the raw byte Array will be returned, in dynamic mode: A Value will be created on the fly. A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.17 Symbol.TryWriteValue Method

Writes the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int TryWriteValue(
    Object value,
    int timeout
)
```

VB

```
Public Function TryWriteValue (
    value As Object,
    timeout As Integer
) As Integer
```

Parameters

value	Type: System.Object The value.
timeout	Type: System.Int32 The timeout in ms.

Return Value

Type: Int32
The error code.

Implements

IValueSymbol3.TryWriteValue(Object, Int32) [▶ 1707]

Remarks

Calling on primitive types, a call of this method will directly write this Value. On complex types (structs and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: Only byte Arrays (of correct size) can be written) in dynamic mode: A Value that represents the value will be accepted also. A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.18 Symbol.UpdateAnyValue Method

Overload List

	Name	Description
	UpdateAnyValue(Object.) [▶ 1091]	Reads the value of this Value [▶ 1683] into the specified managed value.
	UpdateAnyValue(Object., Int32) [▶ 1091]	Reads the value of this Value [▶ 1683] into the specified managed value.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

Symbol.UpdateAnyValue Method (Object.)

Reads the value of this Value [▶ 1683] into the specified managed value.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void UpdateAnyValue(
    ref Object managedObject
)
```

VB

```
Public Sub UpdateAnyValue (
    ByRef managedObject As Object
)
```

Parameters

managedObject	Type: System.Object. The managed object.

Return Value

Type:
Read value (System.Object).

Implements

IValueAnySymbol.UpdateAnyValue(Object.) [▶ 1672]

Reference

Symbol Class [▶ 1057]

UpdateAnyValue Overload [▶ 1090]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

IValueAnySymbol.ReadAnyValue(Type) [▶ 1670]

IValueAnySymbol.WriteAnyValue(Object) [▶ 1673]

Symbol.UpdateAnyValue Method (Object., Int32)

Reads the value of this Value [▶ 1683] into the specified managed value.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void UpdateAnyValue(
    ref Object managedObject,
    int timeout
)
```

VB

```
Public Sub UpdateAnyValue (
    ByRef managedObject As Object,
    timeout As Integer
)
```

Parameters

managedObject	Type: System.Object. The managed object.
timeout	Type: System.Int32 The timeout.

Return Value

Type:
Read value (System.Object).

Implements

IValueAnySymbol.UpdateAnyValue(Object., Int32) [▶ 1672]

Reference

Symbol Class [▶ 1057]

UpdateAnyValue Overload [▶ 1090]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

IValueAnySymbol.ReadAnyValue(Type) [▶ 1670]

IValueAnySymbol.WriteAnyValue(Object) [▶ 1673]

5.5.20.2.19 Symbol.WriteRawValue Method

Overload List

	Name	Description
	WriteRawValue(.Byte.) [▶ 1092]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	WriteRawValue(.Byte., Int32) [▶ 1093]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

Symbol.WriteRawValue Method (.Byte.)

Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteRawValue(
    byte[] value
)
```

VB

```
Public Sub WriteRawValue (
    value As Byte()
)
```

Parameters

value	Type: .System.Byte. The value.

Field Value

Type:
The raw value.

Implements

IValueRawSymbol.WriteRawValue(.Byte.) [▶ 1681]

Reference

Symbol Class [▶ 1057]

WriteRawValue Overload [▶ 1092]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

Symbol.WriteRawValue Method (.Byte., Int32)

Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteRawValue(
    byte[] value,
    int timeout
)
```

VB

```
Public Sub WriteRawValue (
    value As Byte(),
    timeout As Integer
)
```

Parameters

value	Type: .System.Byte. The value.
timeout	Type: System.Int32 The timeout.

Field Value

Type:
The raw value.

Implements

IValueRawSymbol.WriteRawValue(.Byte., Int32) [▶ 1682]

Remarks

A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

Symbol Class [▶ 1057]

WriteRawValue Overload [▶ 1092]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.2.20 Symbol.WriteValue Method

Overload List

	Name	Description
	WriteValue(Object) [▶ 1094]	Writes the Value of the IValueSymbol [▶ 1683]
	WriteValue(Object, Int32) [▶ 1095]	Writes the Value of the IValueSymbol [▶ 1683]

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

Symbol.WriteValue Method (Object)

Writes the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteValue(
    Object value
)
```

VB

```
Public Sub WriteValue (
    value As Object
)
```

Parameters

value	Type: System.Object The value.

Implements

IValueSymbol.WriteValue(Object) [▶ 1692]

Remarks

Calling on primitive types, a call of this method will directly write this Value. On complex types (structs and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: Only byte Arrays (of correct size) can be written) in dynamic mode: A Value that represents the value will be accepted also.

Reference

Symbol Class [▶ 1057]

WriteValue Overload [▶ 1094]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

Symbol.WriteValue Method (Object, Int32)

Writes the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteValue(
    Object value,
    int timeout
)
```

VB

```
Public Sub WriteValue (
    value As Object,
    timeout As Integer
)
```

Parameters

value	Type: System.Object The value.
timeout	Type: System.Int32 The timeout in ms.

Implements

IValueSymbol.WriteValue(Object, Int32) [▶ 1693]

Remarks

Calling on primitive types, a call of this method will directly write this Value. On complex types (structs and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: Only byte Arrays (of correct size) can be written) in dynamic mode: A Value that represents the value will be accepted also. A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

Symbol Class [▶ 1057]

WriteValue Overload [▶ 1094]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.3 Symbol Events

The Symbol [▶ 1057] type exposes the following members.

Events

	Name	Description
⚡	RawValueChanged [▶ 1096]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed.
⚡	ValueChanged [▶ 1097]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.3.1 Symbol.RawValueChanged Event

Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event EventHandler<RawValueChangedArgs> RawValueChanged
```

VB

```
Public Event RawValueChanged As EventHandler(Of RawValueChangedArgs)
```

Value

Type: System.EventHandler.RawValueChangedArgs [▶ 1725].

Implements

IValueRawSymbol.RawValueChanged [▶ 1683]

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.3.2 Symbol.ValueChanged Event

Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event EventHandler<ValueChangedArgs> ValueChanged
```

VB

```
Public Event ValueChanged As EventHandler(Of ValueChangedArgs)
```

Value

Type: System.EventHandler.ValueChangedArgs [▶ 1847].

Implements

IValueSymbol.ValueChanged [▶ 1694]

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.4 Symbol Operators

The Symbol [▶ 1057] type exposes the following members.

Operators

	Name	Description
	Equality [▶ 1098]	Operator==
	Inequality [▶ 1099]	Implements the != operator.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.4.1 Symbol.Equality Operator

Operator==

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator ==(
    Symbol o1,
    Symbol o2
)
```

VB

```
Public Shared Operator = (
    o1 As Symbol,
    o2 As Symbol
) As Boolean
```

Parameters

o1	Type: TwinCAT.Ads.TypeSystem.Symbol [▶ 1057] The o1.
o2	Type: TwinCAT.Ads.TypeSystem.Symbol [▶ 1057] The o2.

Return Value

Type: Boolean
The result of the operator.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.4.2 Symbol.Inequality Operator

Implements the != operator.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator !=(
    Symbol o1,
    Symbol o2
)
```

VB

```
Public Shared Operator <> (
    o1 As Symbol,
    o2 As Symbol
) As Boolean
```

Parameters

o1	Type: TwinCAT.Ads.TypeSystem.Symbol [▶ 1057] The o1.
o2	Type: TwinCAT.Ads.TypeSystem.Symbol [▶ 1057] The o2.

Return Value

Type: Boolean
The result of the operator.

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.20.5 Symbol Fields

The Symbol [▶ 1057] type exposes the following members.

Fields

	Name	Description
	attributes [▶ 1001]	The attributes (Inherited from Instance [▶ 982].)

Reference

Symbol Class [▶ 1057]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.21 SymbolLoaderFactory Class

The class SymbolLoaderFactory is used to create a new instance of the AdsSymbolLoader initialized to the parametrized mode (SymbolBrowser V2, new Version)

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.SymbolLoaderFactory
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SymbolLoaderFactory
```

VB

```
Public Class SymbolLoaderFactory
```

The SymbolLoaderFactory type exposes the following members.

Constructors

	Name	Description
	SymbolLoaderFactory [▶ 1101]	

Methods

	Name	Description
	Create [▶ 1102]	Creates the specified connection.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Remarks

The Symbol Loader (V2) supports the following modes [▶ 108]. Flat [▶ 108]The flat mode organizes the Symbols in a flat list. This mode is available in all .NET versions. VirtualTree [▶ 108]The virtual tree mode organizes the Symbols hierarchically with parent-child relationships. This mode is available in all .NET Versions. DynamicTree [▶ 108]The Dynamic tree mode organizes the Symbols hierarchically and (dynamically) creates struct members, array elements and enum fields on the fly. This feature is only

available on platforms that support the Dynamic Language Runtime (DLR), actually all .NET Framework Version larger than 4.0. Virtual instances means, that all Symbols are ordered within a tree structure. For that symbol nodes that are not located on a fixed address, a Virtual Symbol will be created. Setting the virtualInstance parameter to 'false' means, that the located symbols will be returned in a flattened list.

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

TwinCAT.Ads.SymbolLoaderSettings [▶ 615]

5.5.21.1 SymbolLoaderFactory Constructor

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolLoaderFactory()
```

VB

```
Public Sub New
```

Reference

SymbolLoaderFactory Class [▶ 1100]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.21.2 SymbolLoaderFactory Methods

The SymbolLoaderFactory [▶ 1100] type exposes the following members.

Methods

	Name	Description
	Create [▶ 1102]	Creates the specified connection.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

SymbolLoaderFactory Class [▶ 1100]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.21.2.1 SymbolLoaderFactory.Create Method

Creates the specified connection.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static ISymbolLoader Create(
    IConnection connection,
    ISymbolLoaderSettings settings
)
```

VB

```
Public Shared Function Create (
    connection As IConnection,
    settings As ISymbolLoaderSettings
) As ISymbolLoader
```

Parameters

connection	Type: TwinCAT.IConnection [▶ 46] The connection.
settings	Type: TwinCAT.ISymbolLoaderSettings [▶ 66] The settings.

Return Value

Type: ISymbolLoader [▶ 1645]
ISymbolLoader.

Examples

The following sample shows how to create a dynamic version of the SymbolLoader V2. The dynamic symbol loader makes use of the Dynamic Language Runtime (DLR) of the .NET Framework. That means Structures, Arrays and Enumeration types and instances are generated 'on-the-fly' during symbol Browsing. These created dynamic objects are a one to one representation of the Symbol Server target objects (e.g the IEC61131 types on the PLC). Dynamic language features are only available from .NET4 upwards.

Dynamic Tree Mode

```
namespace Sample
{
    using System;
    using System.Diagnostics;
    using System.Threading;
    using TwinCAT;
    using TwinCAT.Ads;
    using TwinCAT.Ads.TypeSystem;
    using TwinCAT.Ads.ValueAccess;
    using TwinCAT.TypeSystem;
    using TwinCAT.TypeSystem.Generic;

    class SymbolBrowserProgramV2DynamicTree
    {
```

```
    #region CODE_SAMPLE_SIMPLEDYNAMIC
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        // Get the AdsAddress from command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        using (TcAdsClient client = new TcAdsClient())
        {
        // Connect to the target device
        client.Connect(address);

        // Usage of "dynamic" Type and Symbols (>= .NET4 only)
        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.DynamicTree);
        IAdsSymbolLoader dynLoader = (IAdsSymbolLoader)SymbolLoaderFactory.Create(client, settings);

        #endregion

        // Set the Default setting for Notifications
        dynLoader.DefaultNotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 200,
 2000);

        // Get the Symbols (Dynamic Symbols)
        dynamic dynamicSymbols = ((IDynamicSymbolLoader)dynLoader).SymbolsDynamic;

        dynamic adsPort = dynamicSymbols.TwinCAT_SystemInfoVarList._AppInfo.AdsPort;

        #region CODE_SAMPLE_SIMPLEDYNAMIC

        // Access Main Symbol with Dynamic Language Runtime support (DLR)
        // Dynamically created property "Main"
        //dynamic symMain = dynamicSymbols.Main;

        // Main is an 'VirtualSymbol' / Organizational unit that doesn't have a value
        // Calling ReadValue is not allowed
        //bool test = symMain.HasValue;
        //dynamic invalid = symMain.ReadValue();

        //Reading TaskInfo Value
        //With calling ReadValue() a 'snapshot' of the Symbols Instance is taken
        dynamic vTaskInfoArray = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo.ReadValue();

        // Getting the Snapshot time in UTC format
        DateTime timeStamp1 = vTaskInfoArray.UtcTimeStamp;

        // Getting TaskInfo Symbol for Task 1
        dynamic symTaskInfo1 = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo[1];

        // Getting CycleCount Symbol
        dynamic symCycleCount = symTaskInfo1.CycleCount;


        // Take Snapshot value of the ApplicationInfo struct
        dynamic vAppInfo = dynamicSymbols.TwinCAT_SystemInfoVarList._AppInfo.ReadValue();
        // Get the UTC Timestamp of the snapshot

        DateTime timeStamp2 = vAppInfo.UtcTimeStamp;

        // Access the ProjectName of the ApplicationInfo Snapshot (type-safe!)
        string projectNameValue = vAppInfo.ProjectName;

        // Reading the CycleCount Value
        uint cycleCountValue = symTaskInfo1.CycleCount.ReadValue();     // Taking a Value Snapshot
        #endregion

        // Registering for dynamic "ValueChanged" events for the Values
        // Using Default Notification settings
        symCycleCount.ValueChanged += new EventHandler<ValueChangedArgs>(cycleCount_ValueChanged);

        // Override default notification settings
        symTaskInfo1.NotificationSettings = new NotificationSettings(AdsTransMode.Cyclic,500,0);

        // Register for ValueChanged event.
        symTaskInfo1.ValueChanged += new EventHandler<ValueChangedArgs>(taskInfo1Value_ValueChanged)
; // Struct Type

        Thread.Sleep(10000); // Sleep main thread for 10 Seconds
```

```
        }
        Console.WriteLine("CycleCount Changed events received: {0}",_cycleCountEvents);
        Console.WriteLine("taskInfo1 Changed events received: {0}", _taskInfo1Events);

        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
    }

    static object _notificationSynchronizer = new object();
    static int _cycleCountEvents = 0;

    /// <summary>
    /// Handler function for the CycleCount ValueChanged event.
    /// </summary>
    /// <param name="sender">The sender.</param>
    /// <param name="e">The e.</param>
    static void cycleCount_ValueChanged(object sender, ValueChangedArgs e)
    {
        lock(_notificationSynchronizer)
        {
        Interlocked.Increment(ref _cycleCountEvents);
        // val is a type safe value of int!
        dynamic val = e.Value;
        uint intVal = val;

        DateTime changedTime = e.UtcRtime.ToLocalTime(); // Convert UTC to local time
        Console.WriteLine("CycleCount changed to: {0}, TimeStamp: {1}", intVal, changedTime.ToString
("HH:mm:ss:fff"));
        }
    }

    static int _taskInfo1Events = 0;

    /// <summary>
    /// Handler function for the TaskInfo ValueChanged event.
    /// </summary>
    /// <param name="sender">The sender.</param>
    /// <param name="e">The e.</param>
    static void taskInfo1Value_ValueChanged(object sender, ValueChangedArgs e)
    {
        lock (_notificationSynchronizer)
        {
        Interlocked.Increment(ref _taskInfo1Events);
        dynamic val = e.Value;
        DateTime changedTime = e.UtcRtime.ToLocalTime(); // Convert to local time

        // Val is a during Runtime created struct type and contains
        // the same Properties as related PLC object.
        int cycleTime = val.CycleTime;
        Console.WriteLine("TaskInfo1Value changed TimeStamp: {0}", changedTime.ToString("HH:mm:ss:ff
f"));
        }
    }
    }
}
```

The following sample shows how to create a static (non dynamic) version of the SymbolLoader V2. The static symbol loader in version 2 is a nearly code compatible version of the Dynamic Loader, only the dynamic creation of objects is not available. The reason for supporting this mode is that .NET Framework Versions lower than Version 4.0 (CLR2) doesn't support the Dynamic Language Runtime (DLR). The SymbolLoader V2 static object is supported from .NET 2.0 on.

Virtual Tree Mode

```
using System;
using System.Threading;
using System.Diagnostics;
using TwinCAT;
using TwinCAT.Ads;
using TwinCAT.TypeSystem;
using TwinCAT.TypeSystem.Generic;
using TwinCAT.Ads.ValueAccess;
using TwinCAT.Ads.TypeSystem;

namespace Sample
{
    class SymbolBrowserProgramV2VirtualTree
```

```
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        ConsoleLogger logger = new ConsoleLogger();

        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for start:");
        Console.ReadLine();

        //logger.Active = false;

        Stopwatch stopper = new Stopwatch();

        // Parse the command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        stopper.Start();

        using (TcAdsClient client = new TcAdsClient())
        {
        client.Synchronize = false;

        // Connect the AdsClient to the device target.
        client.Connect(address);

        // Creates the Symbol Objects as hierarchical tree
        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.VirtualTree, ValueA
ccessMode.IndexGroupOffsetPreferred);
        ISymbolLoader symbolLoader = SymbolLoaderFactory.Create(client, settings);

        // Dump Datatypes from Target Device
        Console.WriteLine(string.Format("Dumping '{0}' DataTypes:", symbolLoader.DataTypes.Count));
        foreach (IDataType type in symbolLoader.DataTypes)
        {
            logger.DumpType(type);
        }
        Console.WriteLine("");

        // Dump Symbols from target device
        Console.WriteLine("Dumping '{0}' Symbols:", symbolLoader.Symbols.Count);
        foreach (ISymbol symbol in symbolLoader.Symbols)
        {
            logger.DumpSymbol(symbol,0);
        }
        }
        stopper.Stop();
        TimeSpan elapsed = stopper.Elapsed;

        Console.WriteLine("");
        Console.WriteLine("Browsing complete tree: {0},
({1} DataTypes, {2} Symbols)",elapsed,logger.DataTypesCount,logger.DataTypesCount);
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
    }
```

Examples

The SymbolLoader V2 static object is supported from .NET 2.0 on.

Flat Mode

```
using System;
using System.Diagnostics;
using System.Threading;
using TwinCAT;
using TwinCAT.Ads;
using TwinCAT.Ads.TypeSystem;
using TwinCAT.Ads.ValueAccess;
using TwinCAT.TypeSystem;
using TwinCAT.TypeSystem.Generic;

namespace Sample
{
    class SymbolBrowserProgramV2Flat
```

```
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        ConsoleLogger logger = new ConsoleLogger();

        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for start:");
        Console.ReadLine();

        //logger.Active = false;

        Stopwatch stopper = new Stopwatch();

        // Parse the command line arguments
        AmsAddress address = ArgParser.Parse(args);

        stopper.Start();

        // Create the ADS Client
        using (TcAdsClient client = new TcAdsClient())
        {
        client.Synchronize = false;

        // Connect to Address
        client.Timeout = 30000;
        client.Connect(address);

        // Creates the Symbol Objects in Flat Mode (Flat list)
        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.Flat, ValueAccessMo
de.IndexGroupOffsetPreferred);
        ISymbolLoader symbolLoader = SymbolLoaderFactory.Create(client, settings);

        // Dump Datatypes from Target Device
        Console.WriteLine(string.Format("Dumping '{0}' DataTypes:",symbolLoader.DataTypes.Count));
        foreach (IDataType type in symbolLoader.DataTypes)
        {
            logger.DumpType(type);
        }

        Console.WriteLine("");

        // Dump Symbols from target device
        Console.WriteLine("Dumping '{0}' Symbols:",symbolLoader.Symbols.Count);
        foreach (ISymbol symbol in symbolLoader.Symbols)
        {
            logger.DumpSymbol(symbol,0);
        }
        }
        stopper.Stop();
        TimeSpan elapsed = stopper.Elapsed;

        Console.WriteLine("");
        Console.WriteLine("Browsing complete tree: {0},
({1} DataTypes, {2} Symbols)", elapsed, logger.DataTypesCount, logger.DataTypesCount);
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
    }
```

Examples

Argument Parser

```
public static class ArgParser
{
    /// <summary>
    /// Parses the arguments.
    /// </summary>
    /// <param name="args">The arguments.</param>
    /// <returns>AmsAddress.</returns>
    public static AmsAddress Parse(string[] args)
    {
    AmsNetId netId = AmsNetId.Local;
    int port = 851;
```

```
    if (args != null)
    {
        if (args.Length > 0 && args[0] != null)
        netId = AmsNetId.Parse(args[0]);

        if (args.Length > 1 && args[1] != null)
        port = int.Parse(args[1]);
    }
    return new AmsAddress(netId, port);
    }
}
```

Dumping Symbols

```
/// <summary>
/// Console logger
/// </summary>
public class ConsoleLogger
{
    public ConsoleLogger()
    {
    }
    bool _active = true;

    /// <summary>
    /// Gets or sets a value indicating whether this ConsoleLogger is active.
    /// </summary>
    /// <value><c>true</c> if active; otherwise, <c>false</c>.</value>
    public bool Active
    {
    get { return _active; }
    set
    {
        _active = value;
    }
    }

    int _dataTypes = 0;

    /// <summary>
    /// Gets the number of dumped dataTypes.
    /// </summary>
    /// <value>The data types count.</value>
    public int DataTypesCount
    {
    get { return _dataTypes; }
    }

    int _symbols = 0;

    /// <summary>
    /// Gets the number of dumped symbols
    /// </summary>
    /// <value>The symbols count.</value>
    public int SymbolsCount
    {
    get { return _symbols; }
    }


    /// <summary>
    /// Dumps the data type.
    /// </summary>
    /// <param name="dataType">Data Type.</param>
    public void DumpType(IDataType dataType)
    {
    WriteLine(string.Format("DataType: {0}, Category: {1}, Size: {2}", dataType.Name, dataType.Categ
ory, dataType.Size));

    switch (dataType.Category)
    {
        case DataTypeCategory.Alias:
        IAliasType alias = (IAliasType)dataType;
        WriteLine(GetPrefix(1) + string.Format("Alias BaseType: {0}", alias.BaseTypeName));
        break;

        case DataTypeCategory.Enum:

        //IEnumType<ushort> enumType = (IEnumType<ushort>)dataType;
```

```
        IEnumType enumType = (IEnumType)dataType;
        WriteLine(GetPrefix(1) + string.Format("Enum BaseType: {0}", enumType.BaseTypeName));

        foreach (IEnumValue enumValue in enumType.EnumValues)
        {
            WriteLine(GetPrefix(2) + string.Format("Name: {0}, Value: {1}", enumValue.Name, enumValu
e.Primitive));
        }
        break;
        case DataTypeCategory.Array:

        IArrayType arrayType = (IArrayType)dataType;
        int i = 0;

        foreach (IDimension dim in arrayType.Dimensions)
        {
            WriteLine(GetPrefix(2) + string.Format("{0}: LowerBound: {1}, Elements: {2}", i+
+, dim.LowerBound, dim.ElementCount));
        }
        break;
        case DataTypeCategory.Struct:
        IStructType structType = (IStructType)dataType;

        foreach (IMember member in structType.Members)
        {
            WriteLine(GetPrefix(2) + string.Format("Offset {0}: Name: {1}, Type: {2}", member.Offset
, member.InstanceName, member.TypeName));
        }
        break;
        default:
        break;
    }

    foreach (ITypeAttribute attribute in dataType.Attributes)
    {
        WriteLine(GetPrefix(1) + string.Format("{{ {0} : {1} }}", attribute.Name, attribute.Value));
    }
    if (!string.IsNullOrEmpty(dataType.Comment))
    {
        WriteLine(GetPrefix(1) + string.Format("Comment: {0}", dataType.Comment));
    }

    IRpcCallableType rpcCallable = dataType as IRpcCallableType;

    if (rpcCallable != null)
    {
        foreach (IRpcMethod rpcMethod in rpcCallable.RpcMethods)
        {
        if (string.IsNullOrEmpty(rpcMethod.Comment))
            WriteLine(GetPrefix(1) + string.Format("Method: {0}", rpcMethod));
        else
            WriteLine(GetPrefix(1) + string.Format("Method: {0}, Comment: {1}", rpcMethod, rpcMethod
.Comment));
        }
    }
    _dataTypes++;
    }

    /// <summary>
    /// Dumps the Datatype to Console
    /// </summary>
    /// <param name="dataType">DataType.</param>
    public void DumpType(ITcAdsDataType dataType)
    {
    // Dump the Attributes (PLC Metadata)
    foreach (ITypeAttribute attribute in dataType.Attributes)
    {
        WriteLine(GetPrefix(1) + string.Format("{{ {0} : {1} }}", attribute.Name, attribute.Value));
    }

    WriteLine(string.Format("DataType: {0}, Category: {1}, Size: {2}", dataType.Name, dataType.Categ
ory, dataType.Size));

    if (dataType.BaseType != null)
    {
        WriteLine(GetPrefix(1) + string.Format("BaseType: {0}", dataType.BaseType));
    }

    switch (dataType.Category)
    {
```

```
        case DataTypeCategory.Enum:
        foreach (IEnumValue enumValue in dataType.EnumValues)
        {
            WriteLine(GetPrefix(2) + string.Format("Name: {0}, Value: {1}", enumValue.Name, enumValu
e.Primitive));
        }
        break;
        case DataTypeCategory.Array:
        int i = 0;
        foreach (IDimension dim in dataType.Dimensions)
        {
            WriteLine(GetPrefix(2) + string.Format("{0}: LowerBound: {1}, Elements: {2}", i+
+, dim.LowerBound, dim.ElementCount));
        }
        break;
        case DataTypeCategory.Struct:
        foreach (ITcAdsSubItem subItem in dataType.SubItems)
        {
            WriteLine(GetPrefix(2) + string.Format("Offset {0}: Name: {1}, Type: {2}", subItem.Offse
t, subItem.SubItemName, subItem.Name));
        }
        break;
        default:
        break;
    }
    _dataTypes++;
    }

    /// <summary>
    /// Dump Symbol
    /// </summary>
    /// <param name="symbol">The symbol.</param>
    /// <param name="level">Output indentation level</param>
    public void DumpSymbol(ISymbol symbol, int level)
    {
    IDataType type = symbol.DataType as IDataType;

    foreach (ITypeAttribute attribute in symbol.Attributes)
    {
        WriteLine(GetPrefix(level) + string.Format("{{ {0} : {1} }}", attribute.Name, attribute.Valu
e));
    }

       WriteLine(GetPrefix(level) + string.Format("{0} : {1} (IG: 0x{2} IO: 0x{3} size:
{4})", symbol.InstanceName, symbol.TypeName, ((IAdsSymbol)symbol).IndexGroup.ToString("x"), ((IAdsSy
mbol)symbol).IndexOffset.ToString("x"), symbol.Size));

    if (symbol.Category == DataTypeCategory.Array)
    {
        IArrayInstance arrInstance = (IArrayInstance)symbol;
        IArrayType arrType = (IArrayType)symbol.DataType;

        int count = 0;
        level++;

        foreach (ISymbol arrayElement in arrInstance.Elements)
        {
        DumpSymbol(arrayElement, level);
        count++;

        if (count > 20) // Write only the first 20 to limit output
            break;
        }
    }
    else if (symbol.Category == DataTypeCategory.Struct)
    {
        IStructInstance structInstance = (IStructInstance)symbol;
        IStructType structType = (IStructType)symbol.DataType;

        level++;

        foreach (ISymbol member in structInstance.MemberInstances)
        {
        DumpSymbol(member, level);
        }
    }
    _symbols++;
    }

    /// <summary>
```

```
    /// Dumps the specified Symbol to the Console
    /// </summary>
    /// <param name="symbol">The symbol.</param>
    /// <param name="level">The level.</param>
    public void DumpSymbol(ITcAdsSymbol5 symbol, int level)
    {
    // Dump Attributes of the Symbol
    foreach (ITypeAttribute attribute in symbol.Attributes)
    {
        WriteLine(GetPrefix(level) + string.Format("{{ {0} : {1} }}", attribute.Name, attribute.Valu
e));
    }

    ITcAdsSymbolBrowser subSymbolProvider = (ITcAdsSymbolBrowser)symbol;

    // Dump The Symbol
    WriteLine(GetPrefix(level) + string.Format("{0} : {1} ({2}, IG: 0x{3} IO: 0x{4} size:
{6} subCount:
{5})", symbol.Name, symbol.TypeName, symbol.DataTypeId, symbol.IndexGroup.ToString("x"), symbol.Inde
xOffset.ToString("x"), subSymbolProvider.SubSymbols.Count, symbol.Size));
    level++;

    // Dump all SubSymbols with indentation
    foreach (ITcAdsSymbol5 subSymbol in ((ITcAdsSymbolBrowser)symbol).SubSymbols)
    {
        DumpSymbol(subSymbol, level);
    }
    _symbols++;
    }

    /// <summary>
    /// Dump namespace.
    /// </summary>
    /// <param name="ns">The namespace.</param>
    public void DumpNamespace(INamespace<IDataType> ns)
    {
    WriteLine("Namespace: {0}, DataTypes: {1}", ns.Name, ns.DataTypes.Count);

    foreach (IDataType type in ns.DataTypes)
    {
        DumpType(type);
    }
    }

    /// <summary>
    /// Get the indentation prefix
    /// </summary>
    /// <param name="level">The level.</param>
    /// <returns>System.String.</returns>
    public string GetPrefix(int level)
    {
    return "".PadLeft(level * 3);
    }

    /// <summary>
    /// Writes a line to the Console
    /// </summary>
    /// <param name="message">The message.</param>
    public void WriteLine(string message)
    {
    if (Active)
    {
        Console.WriteLine(message);
    }
    }

    /// <summary>
    /// Writes a line to the console
    /// </summary>
    /// <param name="format">The format.</param>
    /// <param name="args">The arguments.</param>
    public void WriteLine(string format, params object[] args)
    {
    if (Active)
    {
        Console.WriteLine(format,args);
    }
    }
}
```

Reference

SymbolLoaderFactory Class [▶ 1100]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.22 UnionType Class

Represents a union type

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.UnionType
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class UnionType : DataType,
    IUnionType, IDataType, IBitSize
```

VB

```
Public NotInheritable Class UnionType
    Inherits DataType
    Implements IUnionType, IDataType, IBitSize
```

The UnionType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	Fields [▶ 1113]	Gets a read only collection of the Fields [▶ 1554] of the IUnionType [▶ 1656].
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)

	Name	Description
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.22.1 UnionType Properties

The UnionType [▶ 1111] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	Fields [▶ 1113]	Gets a read only collection of the Fields [▶ 1554] of the IUnionType [▶ 1656].
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)

	Name	Description
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Reference

UnionType Class [▶ 1111]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.22.1.1 UnionType.Fields Property

Gets a read only collection of the Fields [▶ 1554] of the IUnionType [▶ 1656].

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyFieldCollection Fields { get; }
```

VB

```
Public ReadOnly Property Fields As ReadOnlyFieldCollection
    Get
```

Property Value

Type: ReadOnlyFieldCollection [▶ 1750]
The members as read only collection.

Implements

IUnionType.Fields [▶ 1658]

Reference

UnionType Class [▶ 1111]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.22.2 UnionType Methods

The UnionType [▶ 1111] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 960]	Returns a String that represents this instance. (Inherited from DataType [▶ 945].)

Reference

UnionType Class [▶ 1111]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.23 WStringType Class

Represents an Unicode string (Wide string)

Inheritance Hierarchy

System.Object
  TwinCAT.Ads.TypeSystem.DataType [▶ 945]
    TwinCAT.Ads.TypeSystem.WStringType
Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class WStringType : DataType,
    IStringType, IDataType, IBitSize
```

VB

```
Public NotInheritable Class WStringType
    Inherits DataType
    Implements IStringType, IDataType, IBitSize
```

The WStringType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	Encoding [▶ 1117]	Gets the encoding of the String (Encoding.Default (STRING) or Encoding.UNICODE (WSTRING))
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsFixedLength [▶ 1117]	Gets a value indicating whether the string is of fixed length.
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	Length [▶ 1118]	Gets the number of characters within the string.
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)

	Name	Description
	ToString [▶ 1119]	Returns a String that represents this instance. (Overrides DataType.ToString. [▶ 960].)

Reference

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.23.1 WStringType Properties

The WStringType [▶ 1114] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 948]	Gets the attributes of the IDataType [▶ 1517] (Inherited from DataType [▶ 945].)
	BitSize [▶ 949]	Gets the size of the DataType [▶ 945] in bits. (Inherited from DataType [▶ 945].)
	ByteSize [▶ 950]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DataType [▶ 945].)
	Category [▶ 950]	Gets the Data Type category (Inherited from DataType [▶ 945].)
	Comment [▶ 951]	Gets the comment. (Inherited from DataType [▶ 945].)
	Encoding [▶ 1117]	Gets the encoding of the String (Encoding.Default (STRING) or Encoding.UNICODE (WSTRING))
	FullName [▶ 951]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from DataType [▶ 945].)
	Id [▶ 952]	Gets the ID of the DataType (Inherited from DataType [▶ 945].)
	IsBitType [▶ 952]	Gets a value indicating whether this IDataType [▶ 1517] is a bit mapping Type (Inherited from DataType [▶ 945].)
	IsByteAligned [▶ 953]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DataType [▶ 945].)
	IsContainer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from DataType [▶ 945].)
	IsFixedLength [▶ 1117]	Gets a value indicating whether the string is of fixed length.
	IsPointer [▶ 954]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from DataType [▶ 945].)
	IsPrimitive [▶ 955]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from DataType [▶ 945].)
	IsReference [▶ 956]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from DataType [▶ 945].)
	Length [▶ 1118]	Gets the number of characters within the string.
	ManagedType [▶ 956]	Gets the corresponding .NET Type if attached. (Inherited from DataType [▶ 945].)
	Name [▶ 957]	Gets the name of the Data Type (without namespace) (Inherited from DataType [▶ 945].)

	Name	Description
	Namespace [▶ 957]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from DataType [▶ 945].)
	Size [▶ 958]	Gets the Size of the DataType [▶ 945] in Bytes (Inherited from DataType [▶ 945].)

Reference

WStringType Class [▶ 1114]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.23.1.1 WStringType.Encoding Property

Gets the encoding of the String (Encoding.Default (STRING) or Encoding.UNICODE (WSTRING))

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Encoding Encoding { get; }
```

VB

```
Public ReadOnly Property Encoding As Encoding
    Get
```

Property Value

Type: Encoding
The encoding.

Implements

IStringType.Encoding [▶ 1614]

Reference

WStringType Class [▶ 1114]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.23.1.2 WStringType.IsFixedLength Property

Gets a value indicating whether the string is of fixed length.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsFixedLength { get; }
```

VB

```
Public ReadOnly Property IsFixedLength As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is fixed length; otherwise, false.

Implements

IStringType.IsFixedLength [▶ 1615]

Exceptions

Exception	Condition
NotImplementedException	

Reference

WStringType Class [▶ 1114]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.23.1.3 WStringType.Length Property

Gets the number of characters within the string.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Length { get; }
```

VB

```
Public ReadOnly Property Length As Integer
    Get
```

Property Value

Type: Int32
The length.

Implements

IStringType.Length [▶ 1615]

Reference

WStringType Class [▶ 1114]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.23.2 WStringType Methods

The WStringType [▶ 1114] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString [▶ 1119]	Returns a String that represents this instance. (Overrides DataType.ToString. [▶ 960].)

Reference

WStringType Class [▶ 1114]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.5.23.2.1 WStringType.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.Ads.TypeSystem [▶ 923]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

WStringType Class [▶ 1114]

TwinCAT.Ads.TypeSystem Namespace [▶ 923]

5.6 TwinCAT.Ads.ValueAccess Namespace

Root namespace for ADS value access.

Enumerations

	Enumeration	Description
	ValueAccessMode [▶ 1120]	Enum ValueAccessMethod

5.6.1 ValueAccessMode Enumeration

Enum ValueAccessMethod

Namespace: TwinCAT.Ads.ValueAccess [▶ 1119]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public enum ValueAccessMode
```

VB

```
Public Enumeration ValueAccessMode
```

Members

	Member name	Value	Description
	None	0	None / Uninitialized
	IndexGroupOffset	1	Value access via Index Group and Offset Only
	Symbolic	2	Symbolic access via Instance Path only.
	IndexGroupOffsetPreferred	3	Uses IndexGroup IndexOffset Preferred (and Symbolic for Dereferenced Pointers / References)
	Default	3	The Default access mode (IndexGroupOffsetPreferred)

Reference

TwinCAT.Ads.ValueAccess Namespace [▶ 1119]

5.7 TwinCAT.PlcOpen Namespace

Classes

	Class	Description
	DATE [▶ 1121]	PlcOpen DATE class
	DateBase [▶ 1128]	PlcOpen Date base class.
	LTIME [▶ 1139]	PlcOpen TIME class
	LTimeBase [▶ 1148]	Time base class

	Class	Description
	TIME [▶ 1158]	PlcOpen TIME class
	TimeBase [▶ 1166]	Time base class
	TOD [▶ 1176]	TimeOfDay class

5.7.1 DATE Class

PlcOpen DATE class

Inheritance Hierarchy

System.Object
  TwinCAT.PlcOpen.DateBase [▶ 1128]
    TwinCAT.PlcOpen.DATE

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class DATE : DateBase
```

VB

```
Public NotInheritable Class DATE
    Inherits DateBase
```

The DATE type exposes the following members.

Constructors

	Name	Description
	DATE. [▶ 1122]	Initializes a new instance of the DATE class.
	DATE(DateTime) [▶ 1123]	Initializes a new instance of the DATE class.
	DATE(Int64) [▶ 1123]	Initializes a new instance of the DATE class.
	DATE(UInt32) [▶ 1124]	Initializes a new instance of the DATE class.
	DATE(Int32, Int32, Int32) [▶ 1124]	Initializes a new instance of the DATE class.

Properties

	Name	Description
	Date [▶ 1132]	Gets or sthe date value. (Inherited from DateBase [▶ 1128].)
	Ticks [▶ 1133]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64). (Inherited from DateBase [▶ 1128].)

Methods

	Name	Description
	Equals [▶ 1135]	Determines whether the specified Object is equal to this instance. (Inherited from DateBase [▶ 1128].)
	GetHashCode [▶ 1136]	Gets the HashCode of the Address (Inherited from DateBase [▶ 1128].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
S	Parse [▶ 1126]	Parses the specified string to a DATE object.
	ToString [▶ 1127]	Returns a String that represents this instance. (Overrides Object.ToString..)
S	TryParse [▶ 1127]	Tries to parse the specified string to a DATE object.

Reference

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.1.1 DATE Constructor

Overload List

	Name	Description
	DATE. [▶ 1122]	Initializes a new instance of the DATE [▶ 1121] class.
	DATE(DateTime) [▶ 1123]	Initializes a new instance of the DATE [▶ 1121] class.
	DATE(Int64) [▶ 1123]	Initializes a new instance of the DATE [▶ 1121] class.
	DATE(UInt32) [▶ 1124]	Initializes a new instance of the DATE [▶ 1121] class.
	DATE(Int32, Int32, Int32) [▶ 1124]	Initializes a new instance of the DATE [▶ 1121] class.

Reference

DATE Class [▶ 1121]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.1.1.1 DATE Constructor

Initializes a new instance of the DATE [▶ 1121] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DATE()
```

VB

```
Public Sub New
```

Reference

DATE Class [▶ 1121]

DATE Overload [▶ 1122]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.1.1.2 DATE Constructor (DateTime)

Initializes a new instance of the DATE [▶ 1121] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DATE(
    DateTime date
)
```

VB

```
Public Sub New (
    date As DateTime
)
```

Parameters

date	Type: System.DateTime The date.

Reference

DATE Class [▶ 1121]

DATE Overload [▶ 1122]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.1.1.3 DATE Constructor (Int64)

Initializes a new instance of the DATE [▶ 1121] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DATE(
    long dateValue
)
```

VB

```
Public Sub New (
    dateValue As Long
)
```

Parameters

dateValue	Type: System.Int64 The date value in PlcOpen Ticks.

Reference

DATE Class [▶ 1121]

DATE Overload [▶ 1122]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.1.1.4 DATE Constructor (UInt32)

Initializes a new instance of the DATE [▶ 1121] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DATE(
    uint dateValue
)
```

VB

```
Public Sub New (
    dateValue As UInteger
)
```

Parameters

dateValue	Type: System.UInt32 The date value in PlcOpen Ticks.

Reference

DATE Class [▶ 1121]

DATE Overload [▶ 1122]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.1.1.5 DATE Constructor (Int32, Int32, Int32)

Initializes a new instance of the DATE [▶ 1121] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DATE(
    int year,
    int month,
    int day
)
```

VB

```
Public Sub New (
    year As Integer,
    month As Integer,
    day As Integer
)
```

Parameters

year	Type: System.Int32 The year.
month	Type: System.Int32 The month.
day	Type: System.Int32 The day.

Reference

DATE Class [▶ 1121]

DATE Overload [▶ 1122]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.1.2 DATE Properties

The DATE [▶ 1121] type exposes the following members.

Properties

	Name	Description
	Date [▶ 1132]	Gets or sthe date value. (Inherited from DateBase [▶ 1128].)
	Ticks [▶ 1133]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64). (Inherited from DateBase [▶ 1128].)

Reference

DATE Class [▶ 1121]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.1.3 DATE Methods

The DATE [▶ 1121] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1135]	Determines whether the specified Object is equal to this instance. (Inherited from DateBase [▶ 1128].)
	GetHashCode [▶ 1136]	Gets the HashCode of the Address (Inherited from DateBase [▶ 1128].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
S	Parse [▶ 1126]	Parses the specified string to a DATE [▶ 1121] object.
	ToString [▶ 1127]	Returns a String that represents this instance. (Overrides Object.ToString..)
S	TryParse [▶ 1127]	Tries to parse the specified string to a DATE [▶ 1121] object.

Reference

DATE Class [▶ 1121]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.1.3.1 DATE.Parse Method

Parses the specified string to a DATE [▶ 1121] object.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static DATE Parse(
    string s
)
```

VB

```
Public Shared Function Parse (
    s As String
) As DATE
```

Parameters

s — Type: System.String
The s.

Return Value

Type: DATE [▶ 1121]
DATE.

Exceptions

Exception	Condition
FormatException	Cannot parse DATE object!

Reference

DATE Class [▶ 1121]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.1.3.2 DATE.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

DATE Class [▶ 1121]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.1.3.3 DATE.TryParse Method

Tries to parse the specified string to a DATE [▶ 1121] object.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool TryParse(
    string s,
    out DATE date
)
```

VB

```
Public Shared Function TryParse (
    s As String,
    <OutAttribute> ByRef date As DATE
) As Boolean
```

Parameters

s	Type: System.String The s.

date	Type: TwinCAT.PlcOpen.DATE [▶ 1121]. The date.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

DATE Class [▶ 1121]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2 DateBase Class

PlcOpen Date base class.

Inheritance Hierarchy

System.Object
  TwinCAT.PlcOpen.DateBase
    TwinCAT.PlcOpen.DATE [▶ 1121]
Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public abstract class DateBase : IPlcOpenType<DateTime, uint>,
    IPlcOpenType
```

VB

```
Public MustInherit Class DateBase
    Implements IPlcOpenType(Of DateTime, UInteger),
    IPlcOpenType
```

The DateBase type exposes the following members.

Constructors

	Name	Description
	DateBase. [▶ 1130]	Initializes a new instance of the DateBase class.
	DateBase(DateTime) [▶ 1130]	Initializes a new instance of the DateBase class.
	DateBase(Int64) [▶ 1131]	Initializes a new instance of the DateBase class.
	DateBase(UInt32) [▶ 1131]	Initializes a new instance of the DateBase class.

Properties

	Name	Description
	Date [▶ 1132]	Gets or sthe date value.

	Name	Description
	MarshalSize [▶ 1133]	Gets the marshal size in bytes.
	Ticks [▶ 1133]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64).

Methods

	Name	Description
	DateToValue [▶ 1135]	Converts the specified DateTime value to PlcOpen Ticks.
	Equals [▶ 1135]	Determines whether the specified Object is equal to this instance. (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 1136]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ParseToTicks [▶ 1136]	Parses the specified PlcOpen Date string to PlcOpen ticks.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	ValueToDate(Int64) [▶ 1137]	Converts the specified PlcOpen dateValue in ticks to a DateTime Object
	ValueToDate(UInt32) [▶ 1138]	Converts the specified PlcOpen dateValue in ticks to a DateTime Object

Fields

	Name	Description
	internalDateValue [▶ 1139]	The internal date value

Reference

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.1 DateBase Constructor

Overload List

	Name	Description
	DateBase. [▶ 1130]	Initializes a new instance of the DateBase [▶ 1128] class.
	DateBase(DateTime) [▶ 1130]	Initializes a new instance of the DateBase [▶ 1128] class.

	Name	Description
	DateBase(Int64) [▶ 1131]	Initializes a new instance of the DateBase [▶ 1128] class.
	DateBase(UInt32) [▶ 1131]	Initializes a new instance of the DateBase [▶ 1128] class.

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.1.1 DateBase Constructor

Initializes a new instance of the DateBase [▶ 1128] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected DateBase()
```

VB

```
Protected Sub New
```

Reference

DateBase Class [▶ 1128]

DateBase Overload [▶ 1129]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.1.2 DateBase Constructor (DateTime)

Initializes a new instance of the DateBase [▶ 1128] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected DateBase(
    DateTime date
)
```

VB

```
Protected Sub New (
    date As DateTime
)
```

Parameters

date	Type: System.DateTime The date.

Reference

DateBase Class [▶ 1128]

DateBase Overload [▶ 1129]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.1.3 DateBase Constructor (Int64)

Initializes a new instance of the DateBase [▶ 1128] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected DateBase(
    long dateValue
)
```

VB

```
Protected Sub New (
    dateValue As Long
)
```

Parameters

dateValue	Type: System.Int64 The date value in PlcOpen Ticks.

Exceptions

Exception	Condition
ArgumentOutOfRangeException	

Reference

DateBase Class [▶ 1128]

DateBase Overload [▶ 1129]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.1.4 DateBase Constructor (UInt32)

Initializes a new instance of the DateBase [▶ 1128] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected DateBase(
    uint dateValue
)
```

VB

```
Protected Sub New (
    dateValue As UInteger
)
```

Parameters

dateValue	Type: System.UInt32 The date value in PlcOpen Ticks.

Exceptions

Exception	Condition
ArgumentOutOfRangeException	

Reference

DateBase Class [▶ 1128]

DateBase Overload [▶ 1129]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.2 DateBase Properties

The DateBase [▶ 1128] type exposes the following members.

Properties

	Name	Description
	Date [▶ 1132]	Gets or sthe date value.
S	MarshalSize [▶ 1133]	Gets the marshal size in bytes.
	Ticks [▶ 1133]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64).

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.2.1 DateBase.Date Property

Gets or sthe date value.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DateTime Date { get; }
```

VB

```
Public ReadOnly Property Date As DateTime
    Get
```

Property Value

Type: DateTime
The date.

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.2.2 DateBase.MarshalSize Property

Gets the marshal size in bytes.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static int MarshalSize { get; }
```

VB

```
Public Shared ReadOnly Property MarshalSize As Integer
    Get
```

Property Value

Type: Int32
Marshal size in bytes.

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.2.3 DateBase.Ticks Property

Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64).

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public uint Ticks { get; }
```

VB

```
Public ReadOnly Property Ticks As UInteger
    Get
```

Property Value

Type: UInt32
The ticks.

Implements

IPlcOpenType.Ticks

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.3 DateBase Methods

The DateBase [▶ 1128] type exposes the following members.

Methods

	Name	Description
	DateToValue [▶ 1135]	Converts the specified DateTime value to PlcOpen Ticks.
	Equals [▶ 1135]	Determines whether the specified Object is equal to this instance. (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 1136]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ParseToTicks [▶ 1136]	Parses the specified PlcOpen Date string to PlcOpen ticks.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	ValueToDate(Int64) [▶ 1137]	Converts the specified PlcOpen dateValue in ticks to a DateTime Object
	ValueToDate(UInt32) [▶ 1138]	Converts the specified PlcOpen dateValue in ticks to a DateTime Object

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.3.1 DateBase.DateToValue Method

Converts the specified DateTime value to PlcOpen Ticks.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static long DateToValue(
    DateTime date
)
```

VB

```
Public Shared Function DateToValue (
    date As DateTime
) As Long
```

Parameters

date	Type: System.DateTime The date.

Return Value

Type: Int64

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.3.2 DateBase.Equals Method

Determines whether the specified Object is equal to this instance.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool Equals(
    Object obj
)
```

VB

```
Public Overrides Function Equals (
    obj As Object
) As Boolean
```

Parameters

obj	Type: System.Object The object to compare with the current object.

Return Value

Type: Boolean
true if the specified Object is equal to this instance; otherwise, false.

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.3.3 DateBase.GetHashCode Method

Gets the HashCode of the Address

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override int GetHashCode()
```

VB

```
Public Overrides Function GetHashCode As Integer
```

Return Value

Type: Int32

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.3.4 DateBase.ParseToTicks Method

Parses the specified PlcOpen Date string to PlcOpen ticks.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected abstract long ParseToTicks(
    string s
)
```

VB

```
Protected MustOverride Function ParseToTicks (
    s As String
) As Long
```

Parameters

s	Type: System.String The s.

Return Value

Type: Int64

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.3.5 DateBase.ValueToDate Method

Overload List

	Name	Description
	ValueToDate(Int64) [▶ 1137]	Converts the specified PlcOpen dateValue in ticks to a DateTime Object
	ValueToDate(UInt32) [▶ 1138]	Converts the specified PlcOpen dateValue in ticks to a DateTime Object

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

DateBase.ValueToDate Method (Int64)

Converts the specified PlcOpen dateValue in ticks to a DateTime Object

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static DateTime ValueToDate(
    long dateValue
)
```

VB

```
Public Shared Function ValueToDate (
    dateValue As Long
) As DateTime
```

Parameters

dateValue	Type: System.Int64 The date value.

Return Value

Type: DateTime

Reference

DateBase Class [▶ 1128]

ValueToDate Overload [▶ 1137]

TwinCAT.PlcOpen Namespace [▶ 1120]

DateBase.ValueToDate Method (UInt32)

Converts the specified PlcOpen dateValue in ticks to a DateTime Object

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static DateTime ValueToDate(
    uint dateValue
)
```

VB

```
Public Shared Function ValueToDate (
    dateValue As UInteger
) As DateTime
```

Parameters

dateValue	Type: System.UInt32 The date value.

Return Value

Type: DateTime

Reference

DateBase Class [▶ 1128]

ValueToDate Overload [▶ 1137]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.4 DateBase Fields

The DateBase [▶ 1128] type exposes the following members.

Fields

	Name	Description
	internalDateValue [▶ 1139]	The internal date value

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.2.4.1 DateBase.internalDateValue Field

The internal date value

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected uint internalDateValue
```

VB

```
Protected internalDateValue As UInteger
```

Field Value

Type: UInt32

Reference

DateBase Class [▶ 1128]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3 LTIME Class

PlcOpen TIME class

Inheritance Hierarchy

System.Object
  TwinCAT.PlcOpen.LTimeBase [▶ 1148]
    TwinCAT.PlcOpen.LTIME
Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class LTIME : LTimeBase
```

VB

```
Public NotInheritable Class LTIME
    Inherits LTimeBase
```

The LTIME type exposes the following members.

Constructors

	Name	Description
	LTIME. [▶ 1141]	Initializes a new instance of the TIME [▶ 1158] class.
	LTIME(Int64) [▶ 1142]	Initializes a new instance of the TIME [▶ 1158] class.
	LTIME(TimeSpan) [▶ 1142]	Initializes a new instance of the TIME [▶ 1158] class.
	LTIME(UInt64) [▶ 1143]	Initializes a new instance of the TIME [▶ 1158] class.
	LTIME(Int32, Int32, Int32) [▶ 1143]	Initializes a new instance of the LTIME class.
	LTIME(Int32, Int32, Int32, Int32) [▶ 1144]	Initializes a new instance of the LTIME class.
	LTIME(Int32, Int32, Int32, Int32, Int32, Int32, Int32) [▶ 1145]	Initializes a new instance of the LTIME class.

Properties

	Name	Description
	Ticks [▶ 1152]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64). (Inherited from LTimeBase [▶ 1148].)
	Time [▶ 1153]	Gets or the time value. (Inherited from LTimeBase [▶ 1148].)

Methods

	Name	Description
	Equals [▶ 1154]	Determines whether the specified Object is equal to this instance. (Inherited from LTimeBase [▶ 1148].)
	GetHashCode [▶ 1155]	Gets the HashCode of the Address (Inherited from LTimeBase [▶ 1148].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)

	Name	Description
	Parse [▶ 1147]	Parses the specified string to a LTIME object.
	ToString [▶ 1147]	Returns a String that represents this instance. (Overrides Object.ToString..)
	TryParse [▶ 1148]	Tries to parse the string to a LTIME object.

Reference

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.1 LTIME Constructor

Overload List

	Name	Description
	LTIME. [▶ 1141]	Initializes a new instance of the TIME [▶ 1158] class.
	LTIME(Int64) [▶ 1142]	Initializes a new instance of the TIME [▶ 1158] class.
	LTIME(TimeSpan) [▶ 1142]	Initializes a new instance of the TIME [▶ 1158] class.
	LTIME(UInt64) [▶ 1143]	Initializes a new instance of the TIME [▶ 1158] class.
	LTIME(Int32, Int32, Int32) [▶ 1143]	Initializes a new instance of the LTIME [▶ 1139] class.
	LTIME(Int32, Int32, Int32, Int32) [▶ 1144]	Initializes a new instance of the LTIME [▶ 1139] class.
	LTIME(Int32, Int32, Int32, Int32, Int32, Int32, Int32) [▶ 1145]	Initializes a new instance of the LTIME [▶ 1139] class.

Reference

LTIME Class [▶ 1139]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.1.1 LTIME Constructor

Initializes a new instance of the TIME [▶ 1158] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public LTIME()
```

VB

```
Public Sub New
```

Reference

LTIME Class [▶ 1139]

LTIME Overload [▶ 1141]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.1.2 LTIME Constructor (Int64)

Initializes a new instance of the TIME [▶ 1158] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public LTIME(
    long timeValue
)
```

VB

```
Public Sub New (
    timeValue As Long
)
```

Parameters

timeValue	Type: System.Int64 The time value.

Reference

LTIME Class [▶ 1139]

LTIME Overload [▶ 1141]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.1.3 LTIME Constructor (TimeSpan)

Initializes a new instance of the TIME [▶ 1158] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public LTIME(
    TimeSpan time
)
```

VB

```
Public Sub New (
    time As TimeSpan
)
```

Parameters

time	Type: System.TimeSpan The time.

Reference

LTIME Class [▶ 1139]

LTIME Overload [▶ 1141]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.1.4 LTIME Constructor (UInt64)

Initializes a new instance of the TIME [▶ 1158] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public LTIME(
    ulong timeValue
)
```

VB

```
Public Sub New (
    timeValue As ULong
)
```

Parameters

timeValue	Type: System.UInt64 The time value.

Reference

LTIME Class [▶ 1139]

LTIME Overload [▶ 1141]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.1.5 LTIME Constructor (Int32, Int32, Int32)

Initializes a new instance of the LTIME [▶ 1139] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public LTIME(
    int seconds,
    int milliseconds,
    int microseconds
)
```

VB

```
Public Sub New (
    seconds As Integer,
    milliseconds As Integer,
    microseconds As Integer
)
```

Parameters

seconds	Type: System.Int32 The seconds.
milliseconds	Type: System.Int32 The milliseconds.
microseconds	Type: System.Int32 The microseconds.

Reference

LTIME Class [▶ 1139]

LTIME Overload [▶ 1141]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.1.6 LTIME Constructor (Int32, Int32, Int32, Int32)

Initializes a new instance of the LTIME [▶ 1139] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public LTIME(
    int seconds,
    int milliseconds,
    int microseconds,
    int nanoseconds
)
```

VB

```
Public Sub New (
    seconds As Integer,
    milliseconds As Integer,
    microseconds As Integer,
    nanoseconds As Integer
)
```

Parameters

seconds	Type: System.Int32 The seconds.
milliseconds	Type: System.Int32 The milliseconds.
microseconds	Type: System.Int32 The microseconds.
nanoseconds	Type: System.Int32 The nanoseconds.

Reference

LTIME Class [▶ 1139]

LTIME Overload [▶ 1141]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.1.7 LTIME Constructor (Int32, Int32, Int32, Int32, Int32, Int32, Int32)

Initializes a new instance of the LTIME [▶ 1139] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public LTIME(
    int days,
    int hours,
    int minutes,
    int seconds,
    int milliseconds,
    int microseconds,
    int nanoseconds
)
```

VB

```
Public Sub New (
    days As Integer,
    hours As Integer,
    minutes As Integer,
    seconds As Integer,
    milliseconds As Integer,
    microseconds As Integer,
    nanoseconds As Integer
)
```

Parameters

days	Type: System.Int32 The days.
hours	Type: System.Int32 The hours.
minutes	Type: System.Int32 The minutes.
seconds	Type: System.Int32 The seconds.

milliseconds	Type: System.Int32 The milliseconds.
microseconds	Type: System.Int32 The microseconds.
nanoseconds	Type: System.Int32 The nanoseconds.

Reference

LTIME Class [▶ 1139]

LTIME Overload [▶ 1141]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.2 LTIME Properties

The LTIME [▶ 1139] type exposes the following members.

Properties

	Name	Description
	Ticks [▶ 1152]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64). (Inherited from LTimeBase [▶ 1148].)
	Time [▶ 1153]	Gets or the time value. (Inherited from LTimeBase [▶ 1148].)

Reference

LTIME Class [▶ 1139]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.3 LTIME Methods

The LTIME [▶ 1139] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1154]	Determines whether the specified Object is equal to this instance. (Inherited from LTimeBase [▶ 1148].)
	GetHashCode [▶ 1155]	Gets the HashCode of the Address (Inherited from LTimeBase [▶ 1148].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
 	Parse [▶ 1147]	Parses the specified string to a LTIME [▶ 1139] object.
	ToString [▶ 1147]	Returns a String that represents this instance. (Overrides Object.ToString..)
 	TryParse [▶ 1148]	Tries to parse the string to a LTIME [▶ 1139] object.

Reference

LTIME Class [▶ 1139]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.3.1 LTIME.Parse Method

Parses the specified string to a LTIME [▶ 1139] object.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static LTIME Parse(
    string str
)
```

VB

```
Public Shared Function Parse (
    str As String
) As LTIME
```

Parameters

str	Type: System.String The string.

Return Value

Type: LTIME [▶ 1139]
LTIME.

Exceptions

Exception	Condition
FormatException	Cannot create TIME DataType!

Reference

LTIME Class [▶ 1139]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.3.2 LTIME.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

LTIME Class [▶ 1139]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.3.3.3 LTIME.TryParse Method

Tries to parse the string to a LTIME [▶ 1139] object.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool TryParse(
    string str,
    out LTIME ret
)
```

VB

```
Public Shared Function TryParse (
    str As String,
    <OutAttribute> ByRef ret As LTIME
) As Boolean
```

Parameters

str	Type: System.String The string.
ret	Type: TwinCAT.PlcOpen.LTIME [▶ 1139]. The ret.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

LTIME Class [▶ 1139]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4 LTimeBase Class

Time base class

Inheritance Hierarchy

System.Object
  TwinCAT.PlcOpen.LTimeBase
    TwinCAT.PlcOpen.LTIME [▶ 1139]
Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public abstract class LTimeBase : IPlcOpenType<TimeSpan, ulong>,
    IPlcOpenType
```

VB

```
Public MustInherit Class LTimeBase
    Implements IPlcOpenType(Of TimeSpan, ULong),
    IPlcOpenType
```

The LTimeBase type exposes the following members.

Constructors

	Name	Description
	LTimeBase. [▶ 1150]	Initializes a new instance of the TimeBase [▶ 1166] class.
	LTimeBase(UInt64) [▶ 1151]	Initializes a new instance of the TimeBase [▶ 1166] class.

Properties

	Name	Description
	MarshalSize [▶ 1152]	Gets the marshal size in bytes.
	Ticks [▶ 1152]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64).
	Time [▶ 1153]	Gets or the time value.

Methods

	Name	Description
	Equals [▶ 1154]	Determines whether the specified Object is equal to this instance. (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 1155]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	TimeToValue [▶ 1155]	Converts the Timespan to PlcOpen ticks.

	Name	Description
S		
	ToString	Returns a string that represents the current object. (Inherited from Object.)
S	ValueToTime(Int64) [▶ 1156]	Converts the timeValue (PlcOpen ticks) to TimeSpan
S	ValueToTime(UInt64) [▶ 1157]	Converts the timeValue (PlcOpen ticks) to TimeSpan

Fields

	Name	Description
	internalTimeValue [▶ 1158]	The internal time value

Reference

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.1 LTimeBase Constructor

Overload List

	Name	Description
	LTimeBase. [▶ 1150]	Initializes a new instance of the TimeBase [▶ 1166] class.
	LTimeBase(UInt64) [▶ 1151]	Initializes a new instance of the TimeBase [▶ 1166] class.

Reference

LTimeBase Class [▶ 1148]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.1.1 LTimeBase Constructor

Initializes a new instance of the TimeBase [▶ 1166] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected LTimeBase()
```

VB

```
Protected Sub New
```

Reference

LTimeBase Class [▶ 1148]

LTimeBase Overload [▶ 1150]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.1.2 LTimeBase Constructor (UInt64)

Initializes a new instance of the TimeBase [▶ 1166] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected LTimeBase(
    ulong timeValue
)
```

VB

```
Protected Sub New (
    timeValue As ULong
)
```

Parameters

timeValue	Type: System.UInt64 The time value.

Exceptions

Exception	Condition
ArgumentOutOfRangeException	

Reference

LTimeBase Class [▶ 1148]

LTimeBase Overload [▶ 1150]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.2 LTimeBase Properties

The LTimeBase [▶ 1148] type exposes the following members.

Properties

	Name	Description
	MarshalSize [▶ 1152]	Gets the marshal size in bytes.

	Name	Description
	Ticks [▶ 1152]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64).
	Time [▶ 1153]	Gets or the time value.

Reference

LTimeBase Class [▶ 1148]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.2.1 LTimeBase.MarshalSize Property

Gets the marshal size in bytes.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static int MarshalSize { get; }
```

VB

```
Public Shared ReadOnly Property MarshalSize As Integer
    Get
```

Property Value

Type: Int32
Marshal size in bytes.

Reference

LTimeBase Class [▶ 1148]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.2.2 LTimeBase.Ticks Property

Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64).

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ulong Ticks { get; }
```

VB

```
Public ReadOnly Property Ticks As ULong
    Get
```

Property Value

Type: UInt64
The ticks.

Implements

IPlcOpenType.Ticks

Reference

LTimeBase Class [▶ 1148]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.2.3 LTimeBase.Time Property

Gets or the time value.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TimeSpan Time { get; }
```

VB

```
Public ReadOnly Property Time As TimeSpan
    Get
```

Property Value

Type: TimeSpan
The time.

Reference

LTimeBase Class [▶ 1148]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.3 LTimeBase Methods

The LTimeBase [▶ 1148] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1154]	Determines whether the specified Object is equal to this instance. (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	GetHashCode [▶ 1155]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
S	TimeToValue [▶ 1155]	Converts the Timespan to PlcOpen ticks.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
S	ValueToTime(Int64) [▶ 1156]	Converts the timeValue (PlcOpen ticks) to TimeSpan
S	ValueToTime(UInt64) [▶ 1157]	Converts the timeValue (PlcOpen ticks) to TimeSpan

Reference

LTimeBase Class [▶ 1148]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.3.1 LTimeBase.Equals Method

Determines whether the specified Object is equal to this instance.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool Equals(
    Object obj
)
```

VB

```
Public Overrides Function Equals (
    obj As Object
) As Boolean
```

Parameters

obj	Type: System.Object The object to compare with the current object.

Return Value

Type: Boolean
true if the specified Object is equal to this instance; otherwise, false.

Reference

LTimeBase Class [▶ 1148]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.3.2 LTimeBase.GetHashCode Method

Gets the HashCode of the Address

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override int GetHashCode()
```

VB

```
Public Overrides Function GetHashCode As Integer
```

Return Value

Type: Int32

Reference

LTimeBase Class [▶ 1148]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.3.3 LTimeBase.TimeToValue Method

Converts the Timespan to PlcOpen ticks.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static ulong TimeToValue(
    TimeSpan time
)
```

VB

```
Public Shared Function TimeToValue (
    time As TimeSpan
) As ULong
```

Parameters

time	Type: System.TimeSpan The time.

Return Value

Type: UInt64

Reference

LTimeBase Class [▶ 1148]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.3.4 LTimeBase.ValueToTime Method

Overload List

	Name	Description
	ValueToTime(Int64) [▶ 1156]	Converts the timeValue (PlcOpen ticks) to TimeSpan
	ValueToTime(UInt64) [▶ 1157]	Converts the timeValue (PlcOpen ticks) to TimeSpan

Reference

LTimeBase Class [▶ 1148]

TwinCAT.PlcOpen Namespace [▶ 1120]

LTimeBase.ValueToTime Method (Int64)

Converts the timeValue (PlcOpen ticks) to TimeSpan

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static TimeSpan ValueToTime(
    long nanoseconds
)
```

VB

```
Public Shared Function ValueToTime (
    nanoseconds As Long
) As TimeSpan
```

Parameters

nanoseconds	Type: System.Int64 The time value.

Return Value

Type: TimeSpan

Reference

LTimeBase Class [▶ 1148]

ValueToTime Overload [▶ 1156]

TwinCAT.PlcOpen Namespace [▶ 1120]

LTimeBase.ValueToTime Method (UInt64)

Converts the timeValue (PlcOpen ticks) to TimeSpan

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static TimeSpan ValueToTime(
    ulong nanoseconds
)
```

VB

```
Public Shared Function ValueToTime (
    nanoseconds As ULong
) As TimeSpan
```

Parameters

nanoseconds	Type: System.UInt64 The time value.

Return Value

Type: TimeSpan

Reference

LTimeBase Class [▶ 1148]

ValueToTime Overload [▶ 1156]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.4 LTimeBase Fields

The LTimeBase [▶ 1148] type exposes the following members.

Fields

	Name	Description
	internalTimeValue [▶ 1158]	The internal time value

Reference

LTimeBase Class [▶ 1148]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.4.4.1 LTimeBase.internalTimeValue Field

The internal time value

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected ulong internalTimeValue
```

VB

```
Protected internalTimeValue As ULong
```

Field Value

Type: UInt64

Reference

LTimeBase Class [▶ 1148]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5 TIME Class

PlcOpen TIME class

Inheritance Hierarchy

System.Object
  TwinCAT.PlcOpen.TimeBase [▶ 1166]
    TwinCAT.PlcOpen.TIME
Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class TIME : TimeBase
```

VB

```
Public NotInheritable Class TIME
    Inherits TimeBase
```

The TIME type exposes the following members.

Constructors

	Name	Description
	TIME. [▶ 1160]	Initializes a new instance of the TIME class.
	TIME(Int64) [▶ 1160]	Initializes a new instance of the TIME class.
	TIME(TimeSpan) [▶ 1161]	Initializes a new instance of the TIME class.

	Name	Description
	TIME(UInt32) [▶ 1161]	Initializes a new instance of the TIME class.
	TIME(Int32, Int32) [▶ 1162]	Initializes a new instance of the TIME class.
	TIME(Int32, Int32, Int32, Int32, Int32) [▶ 1162]	Initializes a new instance of the TIME class.

Properties

	Name	Description
	Ticks [▶ 1170]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64). (Inherited from TimeBase [▶ 1166].)
	Time [▶ 1171]	Gets the time value. (Inherited from TimeBase [▶ 1166].)

Methods

	Name	Description
	Equals [▶ 1172]	Determines whether the specified Object is equal to this instance. (Inherited from TimeBase [▶ 1166].)
	GetHashCode [▶ 1173]	Gets the HashCode of the Address (Inherited from TimeBase [▶ 1166].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
S	Parse [▶ 1164]	Parses the specified string to a TIME object.
	ToString [▶ 1165]	Returns a String that represents this instance. (Overrides Object.ToString..)
S	TryParse [▶ 1165]	Tries to parse the TIME object from string.

Reference

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5.1 TIME Constructor

Overload List

	Name	Description
	TIME. [▶ 1160]	Initializes a new instance of the TIME [▶ 1158] class.
	TIME(Int64) [▶ 1160]	Initializes a new instance of the TIME [▶ 1158] class.
	TIME(TimeSpan) [▶ 1161]	Initializes a new instance of the TIME [▶ 1158] class.
	TIME(UInt32) [▶ 1161]	Initializes a new instance of the TIME [▶ 1158] class.
	TIME(Int32, Int32) [▶ 1162]	Initializes a new instance of the TIME [▶ 1158] class.

	Name	Description
	TIME(Int32, Int32, Int32, Int32, Int32) [▶ 1162]	Initializes a new instance of the TIME [▶ 1158] class.

Reference

TIME Class [▶ 1158]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5.1.1 TIME Constructor

Initializes a new instance of the TIME [▶ 1158] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TIME()
```

VB

```
Public Sub New
```

Reference

TIME Class [▶ 1158]

TIME Overload [▶ 1159]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5.1.2 TIME Constructor (Int64)

Initializes a new instance of the TIME [▶ 1158] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TIME(
    long timeValue
)
```

VB

```
Public Sub New (
    timeValue As Long
)
```

Parameters

timeValue	Type: System.Int64 The time value.

Reference

TIME Class [▶ 1158]

TIME Overload [▶ 1159]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5.1.3 TIME Constructor (TimeSpan)

Initializes a new instance of the TIME [▶ 1158] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TIME(
    TimeSpan time
)
```

VB

```
Public Sub New (
    time As TimeSpan
)
```

Parameters

time	Type: System.TimeSpan The time.

Reference

TIME Class [▶ 1158]

TIME Overload [▶ 1159]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5.1.4 TIME Constructor (UInt32)

Initializes a new instance of the TIME [▶ 1158] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TIME(
    uint timeValue
)
```

VB

```
Public Sub New (
    timeValue As UInteger
)
```

Parameters

timeValue	Type: System.UInt32 The time value.

Reference

TIME Class [▶ 1158]

TIME Overload [▶ 1159]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5.1.5 TIME Constructor (Int32, Int32)

Initializes a new instance of the TIME [▶ 1158] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TIME(
    int seconds,
    int milliseconds
)
```

VB

```
Public Sub New (
    seconds As Integer,
    milliseconds As Integer
)
```

Parameters

seconds	Type: System.Int32 The seconds.
milliseconds	Type: System.Int32 The milliseconds.

Reference

TIME Class [▶ 1158]

TIME Overload [▶ 1159]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5.1.6 TIME Constructor (Int32, Int32, Int32, Int32, Int32)

Initializes a new instance of the TIME [▶ 1158] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TIME(
    int days,
    int hours,
    int minutes,
    int seconds,
    int milliseconds
)
```

VB

```
Public Sub New (
    days As Integer,
    hours As Integer,
    minutes As Integer,
    seconds As Integer,
    milliseconds As Integer
)
```

Parameters

days	Type: System.Int32 The days.
hours	Type: System.Int32 The hours.
minutes	Type: System.Int32 The minutes.
seconds	Type: System.Int32 The seconds.
milliseconds	Type: System.Int32 The milliseconds.

Reference

TIME Class [▶ 1158]

TIME Overload [▶ 1159]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5.2 TIME Properties

The TIME [▶ 1158] type exposes the following members.

Properties

	Name	Description
	Ticks [▶ 1170]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64). (Inherited from TimeBase [▶ 1166].)
	Time [▶ 1171]	Gets the time value. (Inherited from TimeBase [▶ 1166].)

Reference

TIME Class [▶ 1158]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5.3 TIME Methods

The TIME [▶ 1158] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1172]	Determines whether the specified Object is equal to this instance. (Inherited from TimeBase [▶ 1166].)
	GetHashCode [▶ 1173]	Gets the HashCode of the Address (Inherited from TimeBase [▶ 1166].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
S	Parse [▶ 1164]	Parses the specified string to a TIME [▶ 1158] object.
	ToString [▶ 1165]	Returns a String that represents this instance. (Overrides Object.ToString..)
S	TryParse [▶ 1165]	Tries to parse the TIME [▶ 1158] object from string.

Reference

TIME Class [▶ 1158]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5.3.1 TIME.Parse Method

Parses the specified string to a TIME [▶ 1158] object.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static TIME Parse(
    string str
)
```

VB

```
Public Shared Function Parse (
    str As String
) As TIME
```

Parameters

str	Type: System.String The string.

Return Value

Type: TIME [▶ 1158]
TIME.

Exceptions

Exception	Condition
FormatException	Cannot create TIME DataType!

Reference

TIME Class [▶ 1158]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5.3.2 TIME.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

TIME Class [▶ 1158]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.5.3.3 TIME.TryParse Method

Tries to parse the TIME [▶ 1158] object from string.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool TryParse(
    string str,
    out TIME ret
)
```

VB

```
Public Shared Function TryParse (
    str As String,
    <OutAttribute> ByRef ret As TIME
) As Boolean
```

Parameters

str	Type: System.String The string.
ret	Type: TwinCAT.PlcOpen.TIME [▶ 1158]. The ret.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

TIME Class [▶ 1158]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6 TimeBase Class

Time base class

Inheritance Hierarchy

System.Object
  TwinCAT.PlcOpen.TimeBase
    TwinCAT.PlcOpen.TIME [▶ 1158]
    TwinCAT.PlcOpen.TOD [▶ 1176]
Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public abstract class TimeBase : IPlcOpenType<TimeSpan, uint>,
    IPlcOpenType
```

VB

```
Public MustInherit Class TimeBase
    Implements IPlcOpenType(Of TimeSpan, UInteger),
    IPlcOpenType
```

The TimeBase type exposes the following members.

Constructors

	Name	Description
	TimeBase. [▶ 1168]	Initializes a new instance of the TimeBase class.
	TimeBase(Int64) [▶ 1168]	Initializes a new instance of the TimeBase class.
	TimeBase(UInt32) [▶ 1169]	Initializes a new instance of the TimeBase class.

Properties

	Name	Description
	MarshalSize [▶ 1170]	Gets the marshal size in bytes.
	Ticks [▶ 1170]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64).
	Time [▶ 1171]	Gets the time value.

Methods

	Name	Description
	Equals [▶ 1172]	Determines whether the specified Object is equal to this instance. (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 1173]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	TimeToValue [▶ 1173]	Converts the Timespan to PlcOpen ticks.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	ValueToTime(Int64) [▶ 1174]	Converts the timeValue (PlcOpen ticks) to TimeSpan
	ValueToTime(UInt32) [▶ 1175]	Converts the timeValue (PlcOpen ticks) to TimeSpan

Fields

	Name	Description
	internalTimeValue [▶ 1176]	The internal time value

Reference

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.1 TimeBase Constructor

Overload List

	Name	Description
	TimeBase. [▶ 1168]	Initializes a new instance of the TimeBase [▶ 1166] class.

	Name	Description
	TimeBase(Int64) [▶ 1168]	Initializes a new instance of the TimeBase [▶ 1166] class.
	TimeBase(UInt32) [▶ 1169]	Initializes a new instance of the TimeBase [▶ 1166] class.

Reference

TimeBase Class [▶ 1166]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.1.1 TimeBase Constructor

Initializes a new instance of the TimeBase [▶ 1166] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected TimeBase()
```

VB

```
Protected Sub New
```

Reference

TimeBase Class [▶ 1166]

TimeBase Overload [▶ 1167]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.1.2 TimeBase Constructor (Int64)

Initializes a new instance of the TimeBase [▶ 1166] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected TimeBase(
    long timeValue
)
```

VB

```
Protected Sub New (
    timeValue As Long
)
```

Parameters

timeValue	Type: System.Int64 The time value.

Exceptions

Exception	Condition
ArgumentOutOfRangeException	

Reference

TimeBase Class [▶ 1166]

TimeBase Overload [▶ 1167]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.1.3 TimeBase Constructor (UInt32)

Initializes a new instance of the TimeBase [▶ 1166] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected TimeBase(
    uint timeValue
)
```

VB

```
Protected Sub New (
    timeValue As UInteger
)
```

Parameters

timeValue	Type: System.UInt32 The time value.

Exceptions

Exception	Condition
ArgumentOutOfRangeException	

Reference

TimeBase Class [▶ 1166]

TimeBase Overload [▶ 1167]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.2 TimeBase Properties

The TimeBase [▶ 1166] type exposes the following members.

Properties

	Name	Description
	MarshalSize [▶ 1170]	Gets the marshal size in bytes.
	Ticks [▶ 1170]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64).
	Time [▶ 1171]	Gets the time value.

Reference

TimeBase Class [▶ 1166]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.2.1 TimeBase.MarshalSize Property

Gets the marshal size in bytes.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static int MarshalSize { get; }
```

VB

```
Public Shared ReadOnly Property MarshalSize As Integer
    Get
```

Property Value

Type: Int32
Marshal size in bytes.

Reference

TimeBase Class [▶ 1166]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.2.2 TimeBase.Ticks Property

Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64).

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public uint Ticks { get; }
```

VB

```
Public ReadOnly Property Ticks As UInteger
    Get
```

Property Value

Type: UInt32
The ticks.

Implements

IPlcOpenType.Ticks

Reference

TimeBase Class [▶ 1166]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.2.3 TimeBase.Time Property

Gets the time value.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual TimeSpan Time { get; }
```

VB

```
Public Overridable ReadOnly Property Time As TimeSpan
    Get
```

Property Value

Type: TimeSpan
The time.

Reference

TimeBase Class [▶ 1166]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.3 TimeBase Methods

The TimeBase [▶ 1166] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1172]	Determines whether the specified Object is equal to this instance. (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 1173]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
S	TimeToValue [▶ 1173]	Converts the Timespan to PlcOpen ticks.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
S	ValueToTime(Int64) [▶ 1174]	Converts the timeValue (PlcOpen ticks) to TimeSpan
S	ValueToTime(UInt32) [▶ 1175]	Converts the timeValue (PlcOpen ticks) to TimeSpan

Reference

TimeBase Class [▶ 1166]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.3.1 TimeBase.Equals Method

Determines whether the specified Object is equal to this instance.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool Equals(
    Object obj
)
```

VB

```
Public Overrides Function Equals (
    obj As Object
) As Boolean
```

Parameters

obj — Type: System.Object
The object to compare with the current object.

Return Value

Type: Boolean
true if the specified Object is equal to this instance; otherwise, false.

Reference

TimeBase Class [▶ 1166]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.3.2 TimeBase.GetHashCode Method

Gets the HashCode of the Address

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override int GetHashCode()
```

VB

```
Public Overrides Function GetHashCode As Integer
```

Return Value

Type: Int32

Reference

TimeBase Class [▶ 1166]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.3.3 TimeBase.TimeToValue Method

Converts the Timespan to PlcOpen ticks.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static long TimeToValue(
    TimeSpan time
)
```

VB

```
Public Shared Function TimeToValue (
    time As TimeSpan
) As Long
```

Parameters

time	Type: System.TimeSpan The time.

Return Value

Type: Int64

Reference

TimeBase Class [▶ 1166]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.3.4 TimeBase.ValueToTime Method

Overload List

	Name	Description
	ValueToTime(Int64) [▶ 1174]	Converts the timeValue (PlcOpen ticks) to TimeSpan
	ValueToTime(UInt32) [▶ 1175]	Converts the timeValue (PlcOpen ticks) to TimeSpan

Reference

TimeBase Class [▶ 1166]

TwinCAT.PlcOpen Namespace [▶ 1120]

TimeBase.ValueToTime Method (Int64)

Converts the timeValue (PlcOpen ticks) to TimeSpan

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static TimeSpan ValueToTime(
    long timeValue
)
```

VB

```
Public Shared Function ValueToTime (
    timeValue As Long
) As TimeSpan
```

Parameters

timeValue	Type: System.Int64 The time value.

Return Value

Type: TimeSpan

Reference

TimeBase Class [▶ 1166]

ValueToTime Overload [▶ 1174]

TwinCAT.PlcOpen Namespace [▶ 1120]

TimeBase.ValueToTime Method (UInt32)

Converts the timeValue (PlcOpen ticks) to TimeSpan

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static TimeSpan ValueToTime(
    uint timeValue
)
```

VB

```
Public Shared Function ValueToTime (
    timeValue As UInteger
) As TimeSpan
```

Parameters

timeValue	Type: System.UInt32 The time value.

Return Value

Type: TimeSpan

Reference

TimeBase Class [▶ 1166]

ValueToTime Overload [▶ 1174]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.4 TimeBase Fields

The TimeBase [▶ 1166] type exposes the following members.

Fields

	Name	Description
	internalTimeValue [▶ 1176]	The internal time value

Reference

TimeBase Class [▶ 1166]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.6.4.1 TimeBase.internalTimeValue Field

The internal time value

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected uint internalTimeValue
```

VB

```
Protected internalTimeValue As UInteger
```

Field Value

Type: UInt32

Reference

TimeBase Class [▶ 1166]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.7 TOD Class

TimeOfDay class

Inheritance Hierarchy

System.Object
  TwinCAT.PlcOpen.TimeBase [▶ 1166]
    TwinCAT.PlcOpen.TOD
Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class TOD : TimeBase
```

VB

```
Public NotInheritable Class TOD
    Inherits TimeBase
```

The TOD type exposes the following members.

Constructors

	Name	Description
	TOD. [▶ 1178]	Initializes a new instance of the TOD class.
	TOD(Int64) [▶ 1178]	Initializes a new instance of the TOD class.
	TOD(TimeSpan) [▶ 1179]	Initializes a new instance of the TOD class.
	TOD(UInt32) [▶ 1180]	Initializes a new instance of the TOD class.
	TOD(Int32, Int32, Int32, Int32, Int32) [▶ 1180]	Initializes a new instance of the TOD class.

Properties

	Name	Description
	Ticks [▶ 1170]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64). (Inherited from TimeBase [▶ 1166].)
	Time [▶ 1171]	Gets the time value. (Inherited from TimeBase [▶ 1166].)

Methods

	Name	Description
	Equals [▶ 1172]	Determines whether the specified Object is equal to this instance. (Inherited from TimeBase [▶ 1166].)
	GetHashCode [▶ 1173]	Gets the HashCode of the Address (Inherited from TimeBase [▶ 1166].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
S	Parse [▶ 1182]	Parses the specified string to a TOD object.
	ToString [▶ 1182]	Returns a string that represents the current object. (Overrides Object.ToString..)
S	TryParse [▶ 1183]	Tries to parse the string to a TOD object.

Reference

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.7.1 TOD Constructor

Overload List

	Name	Description
	TOD. [▶ 1178]	Initializes a new instance of the TOD [▶ 1176] class.
	TOD(Int64) [▶ 1178]	Initializes a new instance of the TOD [▶ 1176] class.
	TOD(TimeSpan) [▶ 1179]	Initializes a new instance of the TOD [▶ 1176] class.
	TOD(UInt32) [▶ 1180]	Initializes a new instance of the TOD [▶ 1176] class.
	TOD(Int32, Int32, Int32, Int32, Int32) [▶ 1180]	Initializes a new instance of the TOD [▶ 1176] class.

Reference

TOD Class [▶ 1176]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.7.1.1 TOD Constructor

Initializes a new instance of the TOD [▶ 1176] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TOD()
```

VB

```
Public Sub New
```

Reference

TOD Class [▶ 1176]

TOD Overload [▶ 1178]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.7.1.2 TOD Constructor (Int64)

Initializes a new instance of the TOD [▶ 1176] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TOD(
    long time
)
```

VB

```
Public Sub New (
    time As Long
)
```

Parameters

time	Type: System.Int64 The time.

Reference

TOD Class [▶ 1176]

TOD Overload [▶ 1178]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.7.1.3 TOD Constructor (TimeSpan)

Initializes a new instance of the TOD [▶ 1176] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TOD(
    TimeSpan timeSpan
)
```

VB

```
Public Sub New (
    timeSpan As TimeSpan
)
```

Parameters

timeSpan	Type: System.TimeSpan The time span.

Reference

TOD Class [▶ 1176]

TOD Overload [▶ 1178]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.7.1.4 TOD Constructor (UInt32)

Initializes a new instance of the TOD [▶ 1176] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TOD(
    uint time
)
```

VB

```
Public Sub New (
    time As UInteger
)
```

Parameters

time	Type: System.UInt32 The time.

Reference

TOD Class [▶ 1176]

TOD Overload [▶ 1178]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.7.1.5 TOD Constructor (Int32, Int32, Int32, Int32, Int32)

Initializes a new instance of the TOD [▶ 1176] class.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TOD(
    int days,
    int hours,
    int minutes,
    int seconds,
    int milliseconds
)
```

VB

```
Public Sub New (
    days As Integer,
    hours As Integer,
    minutes As Integer,
    seconds As Integer,
    milliseconds As Integer
)
```

Parameters

days	Type: System.Int32 The days.
hours	Type: System.Int32 The hours.
minutes	Type: System.Int32 The minutes.
seconds	Type: System.Int32 The seconds.
milliseconds	Type: System.Int32 The milliseconds.

Reference

TOD Class [▶ 1176]

TOD Overload [▶ 1178]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.7.2 TOD Properties

The TOD [▶ 1176] type exposes the following members.

Properties

	Name	Description
	Ticks [▶ 1170]	Returns the number of ticks that represent the value of this IPlcOpenType (uint32 or uint64). (Inherited from TimeBase [▶ 1166].)
	Time [▶ 1171]	Gets the time value. (Inherited from TimeBase [▶ 1166].)

Reference

TOD Class [▶ 1176]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.7.3 TOD Methods

The TOD [▶ 1176] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1172]	Determines whether the specified Object is equal to this instance. (Inherited from TimeBase [▶ 1166].)
	GetHashCode [▶ 1173]	Gets the HashCode of the Address (Inherited from TimeBase [▶ 1166].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
 	Parse [▶ 1182]	Parses the specified string to a TOD [▶ 1176] object.

	Name	Description
	ToString [▶ 1182]	Returns a string that represents the current object. (Overrides Object.ToString..)
S	TryParse [▶ 1183]	Tries to parse the string to a TOD [▶ 1176] object.

Reference

TOD Class [▶ 1176]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.7.3.1 TOD.Parse Method

Parses the specified string to a TOD [▶ 1176] object.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static TOD Parse(
    string str
)
```

VB

```
Public Shared Function Parse (
    str As String
) As TOD
```

Parameters

str	Type: System.String The string.

Return Value

Type: TOD [▶ 1176]
TOD.

Exceptions

Exception	Condition
FormatException	Cannot parse TOD object!

Reference

TOD Class [▶ 1176]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.7.3.2 TOD.ToString Method

Returns a string that represents the current object.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A string that represents the current object.

Reference

TOD Class [▶ 1176]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.7.7.3.3 TOD.TryParse Method

Tries to parse the string to a TOD [▶ 1176] object.

Namespace: TwinCAT.PlcOpen [▶ 1120]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool TryParse(
    string str,
    out TOD ret
)
```

VB

```
Public Shared Function TryParse (
    str As String,
    <OutAttribute> ByRef ret As TOD
) As Boolean
```

Parameters

str	Type: System.String The string.
ret	Type: TwinCAT.PlcOpen.TOD [▶ 1176]. The ret.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

TOD Class [▶ 1176]

TwinCAT.PlcOpen Namespace [▶ 1120]

5.8 TwinCAT.TypeSystem Namespace

Namespace for the common (non ADS dependant) type system.

Classes

	Class	Description
	DataTypeCollection [▶ 1190]	Collection of DataTypes. [▶ 1517]
	DataTypeEventArgs [▶ 1196]	Class DataTypeEventArgs.
	DataTypeException [▶ 1198]	Data Type Exception
	DataTypeNameEventArgs [▶ 1203]	Class DataTypeNameEventArgs.
	Dimension [▶ 1206]	Represents a single dimension of an IArrayType [▶ 1503]
	DimensionCollection [▶ 1209]	Collection class for Array Dimensions
	DynamicAliasInstance [▶ 1227]	Class DynamicAliasInstance. This class cannot be inherited.
	DynamicArrayInstance [▶ 1239]	Dynamic Array Instance
	DynamicOversamplingArrayInstance [▶ 1255]	Dynamic Array Instance
	DynamicPointerInstance [▶ 1264]	Dynamic Pointer Instance
	DynamicPointerValue [▶ 1274]	Class DynamicPointerValue.
	DynamicReferenceInstance [▶ 1281]	Dynamic Reference Instance
	DynamicReferenceValue [▶ 1293]	Class DynamicReferenceValue.
	DynamicRpcStructInstance [▶ 1300]	Dynamic struct instance with RPC Methods.
	DynamicStructInstance [▶ 1315]	Dynamic struct instance

	Class	Description
	DynamicSymbol [▶ 1329]	Dynamic Symbol [▶ 1634] object.
	DynamicSymbolsContainer [▶ 1388]	Dynamic (Expandable) Symbols collection.
	DynamicUnionInstance [▶ 1396]	Dynamic union instance
	DynamicValue [▶ 1407]	Dynamic value (uses RuntimeBinding for ISymbol [▶ 1634] value reading / writing).
	DynamicVirtualStructInstance [▶ 1434]	Dynamic struct instance
	EnumValue.T. [▶ 1443]	Enum Value
	EnumValueCollection [▶ 1450]	Class EnumValueCollection.
	EnumValueCollection.T. [▶ 1467]	Collection of EnumValues [▶ 1443]
	FieldCollection [▶ 1484]	Collection of IField [▶ 1554] objects.
	MarshalException [▶ 1712]	Common Marshalling Exception
	MemberCollection [▶ 1717]	Collection of IMember [▶ 1562] objects.
	RawValueChangedArgs [▶ 1725]	Event args for the RawValueChanged [▶ 1683] event.
	ReadOnlyDataTypeCollection [▶ 1727]	ReadOnly Collection of IDataType [▶ 1517] objects.
	ReadOnlyDimensionCollection [▶ 1731]	ReadOnly version of the DimensionCollection [▶ 1209]
	ReadOnlyEnumValueCollection [▶ 1736]	Read only version of the EnumValueCollection.T. [▶ 1467]
	ReadOnlyEnumValueCollection.T. [▶ 1743]	Read only version of the EnumValueCollection.T. [▶ 1467]
	ReadOnlyFieldCollection [▶ 1750]	Read only collection of IField [▶ 1554] objects

	Class	Description
	ReadOnlyMemberCollection [▶ 1755]	Read only collection of IMember [▶ 1562] objects
	ReadOnlyMethodParameterCollection [▶ 1759]	Read only RpcMethodParameterCollection [▶ 1797].
	ReadOnlyRpcMethodCollection [▶ 1762]	Read only RpcMethodCollection [▶ 1782]
	ReadOnlySubItemCollection [▶ 1768]	Class ReadOnlySubItemCollection.
	ReadOnlySymbolCollection [▶ 1772]	ReadOnly collection containing ISymbol [▶ 1634] objects.
	ReadOnlyTypeAttributeCollection [▶ 1776]	Read only version of the TypeAttributeCollection [▶ 1829]
	RpcMethodCollection [▶ 1782]	Collection of RpcMethods. [▶ 1597]
	RpcMethodParameterCollection [▶ 1797]	Collection of RPC method parameters
	SubItemCollection [▶ 1807]	Class SubItemCollection.
	SymbolCollection [▶ 1818]	Interface represents a collection of ISymbol [▶ 1634] objects.
	TypeAttribute [▶ 1824]	ADS Attribute
	TypeAttributeCollection [▶ 1829]	Collection of AdsAttributes [▶ 1651]
	ValueChangedArgs [▶ 1847]	Event args for the ValueChanged [▶ 1694] event.
	ValueChangedBaseArgs [▶ 1850]	Event args for the RawValueChanged [▶ 1683] event.

Interfaces

	Interface	Description
	IAliasInstance [▶ 1490]	Interface representing an instance of an IAliasType [▶ 1493].
	IAliasType [▶ 1493]	Interface representing an Alias Type
	IArrayInstance [▶ 1496]	Interface representing an array instance

	Interface	Description
	IArrayType [▶ 1503]	Interface representing an array DataType [▶ 1517].
	IArrayValue [▶ 1507]	Interface IArrayValue
	IAttributedInstance [▶ 1512]	Interface IAttributedInstance
	IBitSize [▶ 1514]	Interface IBitSize
	IDataType [▶ 1517]	Base interface for objects representing data types
	IDimension [▶ 1525]	Interface representing a single Dimension [▶ 1525] of an ArrayType [▶ 1503].
	IDimensionCollection [▶ 1527]	Interface IDimensionCollection
	IDynamicSymbol [▶ 1531]	Interface IDynamicSymbol
	IDynamicSymbolLoader [▶ 1535]	Dynamic symbol loader interface
	IEnumType [▶ 1536]	Common Enum type interface
	IEnumType.T. [▶ 1543]	Interface representing an enum type
	IEnumValue [▶ 1550]	Generic interface for EnumValues
	IField [▶ 1554]	Specifies a single field/member of a Struct DataType [▶ 1619].
	IInstance [▶ 1556]	Interface specifying instance objects.
	IMember [▶ 1562]	Specifies a single field/member of a Struct DataType [▶ 1619].
	INotificationSettings [▶ 1566]	Interface for Notification Settings
	IOversamplingArrayInstance [▶ 1567]	Interface IOversamplingArrayInstance
	IPointerInstance [▶ 1571]	Interface representing an instance of an IPointerType [▶ 1574]
	IPointerType [▶ 1574]	Interface representing a pointer type
	IPrimitiveType [▶ 1577]	Interface IPrimitiveType
	IProcessImageAddress [▶ 1579]	Interface describing a Process Image Address
	IReferenceInstance [▶ 1582]	Interface representing an instance of an IReferenceType [▶ 1587]
	IReferenceType [▶ 1587]	Interface representing a reference/pointer type
	IRpcCallableInstance [▶ 1591]	Interface for an RPC callable PLC Method (Remote procedure call)

	Interface	Description
	IRpcCallableType [▶ 1596]	Interface representing an RPC callable IStructType [▶ 1619]
	IRpcMethod [▶ 1597]	Interface describes an RPC Method
	IRpcMethodParameter [▶ 1601]	Interface IRpcMethodParameter
	IRpcStructInstance [▶ 1604]	Interface IRpcStructInstance
	IStringInstance [▶ 1608]	Interface IStringInstance
	IStringType [▶ 1612]	Interface representing a string IDataType [▶ 1517]
	IStructInstance [▶ 1615]	Interface representing an instance of a IStructType [▶ 1619]
	IStructType [▶ 1619]	Interface representing Struct data types
	IStructValue [▶ 1624]	Interface IStructValue
	ISubRangeType [▶ 1628]	Interface representing a SubRange type
	ISubRangeType.T. [▶ 1630]	Interface representing a SubRange type
	ISymbol [▶ 1634]	Interface specifying Symbols (
	ISymbolCollection [▶ 1640]	Interface ISymbolCollection
	ISymbolInfo [▶ 1643]	Interface ISymbolInfo
	ISymbolLoader [▶ 1645]	Symbol Loader interface
	ISymbolProvider [▶ 1647]	Symbol Provider interface.
	ISymbolServer [▶ 1649]	Symbol Server Interface
	ITypeAttribute [▶ 1651]	Interface for ADS attributes
	IUnionInstance [▶ 1653]	Interface for an Instance of the IUnionType [▶ 1656].
	IUnionType [▶ 1656]	Interface for an union data type.
	IValue [▶ 1659]	Symbol Value Interface
	IValueAnySymbol [▶ 1666]	Interface IValueAnySymbol
	IValueRawSymbol [▶ 1675]	Interface IValueRawSymbol
	IValueSymbol [▶ 1683]	Interface for a ISymbol [▶ 1634] that supports values.
	IValueSymbol2 [▶ 1694]	Interface for a ISymbol [▶ 1634] that supports values.

	Interface	Description
	IValueSymbol3 [▶ 1701]	Interface IValueSymbol3 Implements the IValueSymbol2 [▶ 1694]
	IVirtualStructInstance [▶ 1708]	Virtual Struct instance interface.

Enumerations

	Enumeration	Description
	DataTypeCategory [▶ 1189]	Category of a DataType / Instance
	MethodParamFlags [▶ 1724]	Flag set specifying the MethodParameter context
	PrimitiveTypeFlags [▶ 1724]	Enum PrimitiveTypeFlags
	SymbolAccessRights [▶ 1818]	Enum specifying Access Rights to symbols

5.8.1 DataTypeCategory Enumeration

Category of a DataType / Instance

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public enum DataTypeCategory
```

VB

```
Public Enumeration DataTypeCategory
```

Members

	Member name	Value	Description
	Unknown	0	Uninitialized / NotProcessed (0)
	None	0	Uninitialized / NotProcessed (0)
	Primitive	1	Simple / Base Data Type (1)
	Alias	2	Alias data type (2)
	Enum	3	Enumeration data type (3)
	Array	4	Array data type (4)
	Struct	5	Structure data type (5)
	FunctionBlock	6	Function block (POU) (6)
	Program	7	Program (POU) (7)
	Function	8	Function (POU) (8)
	SubRange	9	SubRange (9)
	String	10	Fixed length string (10)
	Bitset	12	Bitset (12)
	Pointer	13	Pointer type (13)
	Union	14	Union type (14)

	Member name	Value	Description
	Reference	15	Reference type (15)
	Interface	16	The interface

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.2 DataTypeCollection Class

Collection of DataTypes. [▶ 1517]

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.Generic.DataTypeCollection [▶ 1855].IDataType [▶ 1517].
    TwinCAT.TypeSystem.DataTypeCollection
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class DataTypeCollection : DataTypeCollection<IDataType>
```

VB

```
Public Class DataTypeCollection
    Inherits DataTypeCollection(Of IDataType)
```

The DataTypeCollection type exposes the following members.

Constructors

	Name	Description
	DataTypeCollection. [▶ 1192]	Initializes a new instance of the DataTypeCollection class.
	DataTypeCollection(IEnumerable.IDataType.) [▶ 1192]	Initializes a new instance of the DataTypeCollection class (Copy constructor).

Properties

	Name	Description
	Count [▶ 1859]	Gets the count of contained IDataType [▶ 1517]s. (Inherited from DataTypeCollection.T. [▶ 1855].)
	IsReadOnly [▶ 1859]	Gets a value indicating whether this instance is read only. (Inherited from DataTypeCollection.T. [▶ 1855].)
	Item.Int32. [▶ 1860]	Gets or sets the IDataType [▶ 1517] at the specified index. (Inherited from DataTypeCollection.T. [▶ 1855].)
	Item.String. [▶ 1861]	Gets the IDataType [▶ 1517] with the specified name. (Inherited from DataTypeCollection.T. [▶ 1855].)

Methods

	Name	Description
	Add [▶ 1863]	Adds the specified item to the collection. (Inherited from DataTypeCollection.T. [▶ 1855].)
	AddRange [▶ 1863]	Adds a range of types (Inherited from DataTypeCollection.T. [▶ 1855].)
	AsReadOnly [▶ 1194]	Returns A ReadOnly-Version of the DataTypeCollection.
	Clear [▶ 1864]	Clears the collection. (Inherited from DataTypeCollection.T. [▶ 1855].)
	Clone [▶ 1195]	Clones this DataTypeCollection (Shallow Copy)
	Contains [▶ 1865]	Determines whether this DataTypeCollection contains the specified IDataType [▶ 1517]. (Inherited from DataTypeCollection.T. [▶ 1855].)
	ContainsType [▶ 1866]	Determines whether the container contains the specified IDataType [▶ 1517]. (Inherited from DataTypeCollection.T. [▶ 1855].)
	CopyTo [▶ 1867]	Copies the data types to the specified array, starting at the array index. (Inherited from DataTypeCollection.T. [▶ 1855].)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1867]	Gets the enumerator. (Inherited from DataTypeCollection.T. [▶ 1855].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1868]	Determines the Index of the specified IDataType [▶ 1517]. (Inherited from DataTypeCollection.T. [▶ 1855].)
	Insert [▶ 1868]	Inserts an IDataType [▶ 1517] into the DataTypeCollection. (Inherited from DataTypeCollection.T. [▶ 1855].)
	LookupType [▶ 1869]	Determines the specified IDataType [▶ 1517] (Inherited from DataTypeCollection.T. [▶ 1855].)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1870]	Removes the specified IDataType [▶ 1517]. (Inherited from DataTypeCollection.T. [▶ 1855].)
	RemoveAt [▶ 1870]	Removes the IDataType [▶ 1517] object at the specified index. (Inherited from DataTypeCollection.T. [▶ 1855].)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetType [▶ 1871]	Tries to get the specified IDataType [▶ 1517] from the IDataTypeContainer.T. [▶ 1874]. (Inherited from DataTypeCollection.T. [▶ 1855].)

Fields

	Name	Description
	list [▶ 1872]	Internal list of data types (Inherited from DataTypeCollection.T. [▶ 1855].)
	nameDict [▶ 1873]	Dictionary (Type Name --> DataType) (Inherited from DataTypeCollection.T. [▶ 1855].)
	readOnly [▶ 1873]	Indicates that the DataTypeCollection.T. [▶ 1855] is readonly (Inherited from DataTypeCollection.T. [▶ 1855].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.2.1 DataTypeCollection Constructor

Overload List

	Name	Description
	DataTypeCollection. [▶ 1192]	Initializes a new instance of the DataTypeCollection [▶ 1190] class.
	DataTypeCollection(IEnumerable.IDataType.) [▶ 1192]	Initializes a new instance of the DataTypeCollection [▶ 1190] class (Copy constructor).

Reference

DataTypeCollection Class [▶ 1190]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.2.1.1 DataTypeCollection Constructor

Initializes a new instance of the DataTypeCollection [▶ 1190] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeCollection()
```

VB

```
Public Sub New
```

Reference

DataTypeCollection Class [▶ 1190]

DataTypeCollection Overload [▶ 1192]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.2.1.2 DataTypeCollection Constructor (IEnumerable.IDataType.)

Initializes a new instance of the DataTypeCollection [▶ 1190] class (Copy constructor).

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeCollection(
    IEnumerable<IDataType> coll
)
```

VB

```
Public Sub New (
    coll As IEnumerable(Of IDataType)
)
```

Parameters

coll	Type: System.Collections.Generic.IEnumerable.IDataType [▶ 1517]. The coll.

Reference

DataTypeCollection Class [▶ 1190]

DataTypeCollection Overload [▶ 1192]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.2.2 DataTypeCollection Properties

The DataTypeCollection [▶ 1190] type exposes the following members.

Properties

	Name	Description
	Count [▶ 1859]	Gets the count of contained IDataType [▶ 1517]s. (Inherited from DataTypeCollection.T. [▶ 1855].)
	IsReadOnly [▶ 1859]	Gets a value indicating whether this instance is read only. (Inherited from DataTypeCollection.T. [▶ 1855].)
	Item.Int32. [▶ 1860]	Gets or sets the IDataType [▶ 1517] at the specified index. (Inherited from DataTypeCollection.T. [▶ 1855].)
	Item.String. [▶ 1861]	Gets the IDataType [▶ 1517] with the specified name. (Inherited from DataTypeCollection.T. [▶ 1855].)

Reference

DataTypeCollection Class [▶ 1190]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.2.3 DataTypeCollection Methods

The DataTypeCollection [▶ 1190] type exposes the following members.

Methods

	Name	Description
	Add [▶ 1863]	Adds the specified item to the collection. (Inherited from DataTypeCollection.T. [▶ 1855].)

	Name	Description
	AddRange [▶ 1863]	Adds a range of types (Inherited from DataTypeCollection.T. [▶ 1855].)
	AsReadOnly [▶ 1194]	Returns A ReadOnly-Version of the DataTypeCollection [▶ 1190].
	Clear [▶ 1864]	Clears the collection. (Inherited from DataTypeCollection.T. [▶ 1855].)
	Clone [▶ 1195]	Clones this DataTypeCollection [▶ 1190] (Shallow Copy)
	Contains [▶ 1865]	Determines whether this DataTypeCollection [▶ 1190] contains the specified IDataType [▶ 1517]. (Inherited from DataTypeCollection.T. [▶ 1855].)
	ContainsType [▶ 1866]	Determines whether the container contains the specified IDataType [▶ 1517]. (Inherited from DataTypeCollection.T. [▶ 1855].)
	CopyTo [▶ 1867]	Copies the data types to the specified array, starting at the array index. (Inherited from DataTypeCollection.T. [▶ 1855].)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1867]	Gets the enumerator. (Inherited from DataTypeCollection.T. [▶ 1855].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1868]	Determines the Index of the specified IDataType [▶ 1517]. (Inherited from DataTypeCollection.T. [▶ 1855].)
	Insert [▶ 1868]	Inserts an IDataType [▶ 1517] into the DataTypeCollection [▶ 1190]. (Inherited from DataTypeCollection.T. [▶ 1855].)
	LookupType [▶ 1869]	Determines the specified IDataType [▶ 1517] (Inherited from DataTypeCollection.T. [▶ 1855].)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1870]	Removes the specified IDataType [▶ 1517]. (Inherited from DataTypeCollection.T. [▶ 1855].)
	RemoveAt [▶ 1870]	Removes the IDataType [▶ 1517] object at the specified index. (Inherited from DataTypeCollection.T. [▶ 1855].)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetType [▶ 1871]	Tries to get the specified IDataType [▶ 1517] from the IDataTypeContainer.T. [▶ 1874]. (Inherited from DataTypeCollection.T. [▶ 1855].)

Reference

DataTypeCollection Class [▶ 1190]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.2.3.1 DataTypeCollection.AsReadOnly Method

Returns A ReadOnly-Version of the DataTypeCollection [▶ 1190].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyDataTypeCollection AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlyDataTypeCollection
```

Return Value

Type: ReadOnlyDataTypeCollection [▶ 1727]
A read only version of this DataTypeCollection [▶ 1190].

Reference

DataTypeCollection Class [▶ 1190]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.2.3.2 DataTypeCollection.Clone Method

Clones this DataTypeCollection [▶ 1190] (Shallow Copy)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeCollection Clone()
```

VB

```
Public Function Clone As DataTypeCollection
```

Return Value

Type: DataTypeCollection [▶ 1190]
A clone of this DataTypeCollection [▶ 1190].

Reference

DataTypeCollection Class [▶ 1190]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.2.4 DataTypeCollection Fields

The DataTypeCollection [▶ 1190] type exposes the following members.

Fields

	Name	Description
	list [▶ 1872]	Internal list of data types (Inherited from DataTypeCollection.T. [▶ 1855].)

	Name	Description
	nameDict [▶ 1873]	Dictionary (Type Name --> DataType) (Inherited from DataTypeCollection.T. [▶ 1855].)
	readOnly [▶ 1873]	Indicates that the DataTypeCollection.T. [▶ 1855] is readonly (Inherited from DataTypeCollection.T. [▶ 1855].)

Reference

DataTypeCollection Class [▶ 1190]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.3 DataTypeEventArgs Class

Class DataTypeEventArgs.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.TypeSystem.DataTypeEventArgs
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class DataTypeEventArgs : EventArgs
```

VB

```
Public Class DataTypeEventArgs
    Inherits EventArgs
```

The DataTypeEventArgs type exposes the following members.

Constructors

	Name	Description
	DataTypeEventArgs [▶ 1197]	Initializes a new instance of the DataTypeEventArgs class.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Fields

	Name	Description
	DataTypes [▶ 1198]	The data types

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

System.EventArgs

5.8.3.1 DataTypeEventArgs Constructor

Initializes a new instance of the DataTypeEventArgs [▶ 1196] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeEventArgs(
    IEnumerable<IDataType> types
)
```

VB

```
Public Sub New (
    types As IEnumerable(Of IDataType)
)
```

Parameters

types	Type: System.Collections.Generic.IEnumerable.IDataType [▶ 1517]. The types.

Reference

DataTypeEventArgs Class [▶ 1196]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.3.2 DataTypeEventArgs Methods

The DataTypeEventArgs [▶ 1196] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)

	Name	Description
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

DataTypeEventArgs Class [▶ 1196]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.3.3 DataTypeEventArgs Fields

The DataTypeEventArgs [▶ 1196] type exposes the following members.

Fields

	Name	Description
	DataTypes [▶ 1198]	The data types

Reference

DataTypeEventArgs Class [▶ 1196]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.3.3.1 DataTypeEventArgs.DataTypes Field

The data types

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly IEnumerable<IDataType> DataTypes
```

VB

```
Public ReadOnly DataTypes As IEnumerable(Of IDataType)
```

Field Value

Type: IEnumerable.IDataType [▶ 1517].

Reference

DataTypeEventArgs Class [▶ 1196]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.4 DataTypeException Class

Data Type Exception

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.TypeSystem.DataTypeException

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class DataTypeException : AdsException
```

VB

```
<SerializableAttribute>
Public Class DataTypeException
    Inherits AdsException
```

The DataTypeException type exposes the following members.

Constructors

	Name	Description
	DataTypeException [▶ 1200]	Initializes a new instance of the DataTypeException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)

	Name	Description
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Fields

	Name	Description
	DataType [▶ 1203]	The data type

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.4.1 DataTypeException Constructor

Initializes a new instance of the DataTypeException [▶ 1198] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeException(
    string message,
    IDataType type
)
```

VB

```
Public Sub New (
    message As String,
    type As IDataType
)
```

Parameters

message	Type: System.String The message.
type	Type: TwinCAT.TypeSystem.IDataType [▶ 1517] The type.

Reference

DataTypeException Class [▶ 1198]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.4.2 DataTypeException Properties

The DataTypeException [▶ 1198] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

DataTypeException Class [▶ 1198]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.4.3 DataTypeException Methods

The DataTypeException [▶ 1198] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)

	Name	Description
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

DataTypeException Class [▶ 1198]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.4.4 DataTypeException Events

The DataTypeException [▶ 1198] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

DataTypeException Class [▶ 1198]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.4.5 DataTypeException Fields

The DataTypeException [▶ 1198] type exposes the following members.

Fields

	Name	Description
	DataType [▶ 1203]	The data type

Reference

DataTypeException Class [▶ 1198]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.4.5.1 DataTypeException.DataType Field

The data type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[NonSerializedAttribute]
public readonly IDataType DataType
```

VB

```
<NonSerializedAttribute>
Public ReadOnly DataType As IDataType
```

Field Value

Type: IDataType [▶ 1517]

Reference

DataTypeException Class [▶ 1198]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.5 DataTypeNameEventArgs Class

Class DataTypeNameEventArgs.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.TypeSystem.DataTypeNameEventArgs
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class DataTypeNameEventArgs : EventArgs
```

VB

```
Public Class DataTypeNameEventArgs
    Inherits EventArgs
```

The DataTypeNameEventArgs type exposes the following members.

Constructors

	Name	Description
	DataTypeNameEventArgs [▶ 1204]	Initializes a new instance of the DataTypeNameEventArgs class.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Fields

	Name	Description
	TypeName [▶ 1205]	The type name

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

System.EventArgs

5.8.5.1 DataTypeNameEventArgs Constructor

Initializes a new instance of the DataTypeNameEventArgs [▶ 1203] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeNameEventArgs(
    string typeName
)
```

VB

```
Public Sub New (
    typeName As String
)
```

Parameters

typeName	Type: System.String Name of the type.

Reference

DataTypeNameEventArgs Class [▶ 1203]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.5.2 DataTypeNameEventArgs Methods

The DataTypeNameEventArgs [▶ 1203] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

DataTypeNameEventArgs Class [▶ 1203]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.5.3 DataTypeNameEventArgs Fields

The DataTypeNameEventArgs [▶ 1203] type exposes the following members.

Fields

	Name	Description
	TypeName [▶ 1205]	The type name

Reference

DataTypeNameEventArgs Class [▶ 1203]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.5.3.1 DataTypeNameEventArgs.TypeName Field

The type name

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly string TypeName
```

VB

```
Public ReadOnly TypeName As String
```

Field Value

Type: String

Reference

DataTypeNameEventArgs Class [▶ 1203]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.6 Dimension Class

Represents a single dimension of an IArrayType [▶ 1503]

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.Dimension
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class Dimension : IDimension
```

VB

```
Public Class Dimension
    Implements IDimension
```

The Dimension type exposes the following members.

Properties

	Name	Description
	ElementCount [▶ 1207]	Gets the number of elements within that IDimension [▶ 1525].
	LowerBound [▶ 1208]	Gets the lower bound of elements within that IDimension [▶ 1525].
	UpperBound [▶ 1208]	Gets the upper bound of elements within this Dimension

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.6.1 Dimension Properties

The Dimension [▶ 1206] type exposes the following members.

Properties

	Name	Description
	ElementCount [▶ 1207]	Gets the number of elements within that IDimension [▶ 1525].
	LowerBound [▶ 1208]	Gets the lower bound of elements within that IDimension [▶ 1525].
	UpperBound [▶ 1208]	Gets the upper bound of elements within this Dimension [▶ 1206]

Reference

Dimension Class [▶ 1206]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.6.1.1 Dimension.ElementCount Property

Gets the number of elements within that IDimension [▶ 1525].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ElementCount { get; }
```

VB

```
Public ReadOnly Property ElementCount As Integer
    Get
```

Property Value

Type: Int32
The element count.

Implements

IDimension.ElementCount [▶ 1526]

Reference

Dimension Class [▶ 1206]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.6.1.2 Dimension.LowerBound Property

Gets the lower bound of elements within that IDimension [▶ 1525].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int LowerBound { get; }
```

VB

```
Public ReadOnly Property LowerBound As Integer
    Get
```

Property Value

Type: Int32
The lower bound.

Implements

IDimension.LowerBound [▶ 1527]

Reference

Dimension Class [▶ 1206]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.6.1.3 Dimension.UpperBound Property

Gets the upper bound of elements within this Dimension [▶ 1206]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int UpperBound { get; }
```

VB

```
Public ReadOnly Property UpperBound As Integer
    Get
```

Property Value

Type: Int32
The upper bound.

Reference

Dimension Class [▶ 1206]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.6.2 Dimension Methods

The Dimension [▶ 1206] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

Dimension Class [▶ 1206]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7 DimensionCollection Class

Collection class for Array Dimensions

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.DimensionCollection
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class DimensionCollection : IDimensionCollection,
    IList<IDimension>, ICollection<IDimension>, IEnumerable<IDimension>,
    IEnumerable
```

VB

```
Public Class DimensionCollection
    Implements IDimensionCollection, IList(Of IDimension),
    ICollection(Of IDimension), IEnumerable(Of IDimension), IEnumerable
```

The DimensionCollection type exposes the following members.

Constructors

	Name	Description
	DimensionCollection. [▶ 1211]	Initializes a new instance of the DimensionCollection class.

	Name	Description
	DimensionCollection(Int32) [▶ 1212]	Initializes a new instance of an 1-Dimensional representing DimensionCollection class.
	DimensionCollection(.Int32.) [▶ 1212]	Initializes a new instance of the DimensionCollection class.
	DimensionCollection(IEnumerable.IDimension.) [▶ 1213]	Initializes a new instance of the DimensionCollection class.

Properties

	Name	Description
	Count [▶ 1214]	Gets the number of elements contained in the ICollection.T..
	ElementCount [▶ 1215]	Gets the Number of elements in all Dimensions
	IsReadOnly [▶ 1215]	Gets a value indicating whether the ICollection.T. is read-only.
	Item [▶ 1216]	Gets or sets the element at the specified index.
	LowerBounds [▶ 1217]	Gets the lower bounds.
	UpperBounds [▶ 1217]	Gets the upper bounds.

Methods

	Name	Description
	Add [▶ 1219]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1219]	Returns a read only version of this DimensionCollection.
	Clear [▶ 1220]	Removes all items from the ICollection.T..
	Contains [▶ 1221]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1221]	Copies to.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDimensionLengths [▶ 1222]	Gets an array the specifies the Lengths of each Array Dimension
	GetEnumerator [▶ 1223]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1223]	Determines the index of a specific item in the IList.T..

	Name	Description
	Insert [▶ 1224]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1225]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1226]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.1 DimensionCollection Constructor

Overload List

	Name	Description
	DimensionCollection. [▶ 1211]	Initializes a new instance of the DimensionCollection [▶ 1209] class.
	DimensionCollection(Int32) [▶ 1212]	Initializes a new instance of an 1-Dimensional representing DimensionCollection [▶ 1209] class.
	DimensionCollection(.Int32.) [▶ 1212]	Initializes a new instance of the DimensionCollection [▶ 1209] class.
	DimensionCollection(IEnumerable.IDimension.) [▶ 1213]	Initializes a new instance of the DimensionCollection [▶ 1209] class.

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.1.1 DimensionCollection Constructor

Initializes a new instance of the DimensionCollection [▶ 1209] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DimensionCollection()
```

VB

```
Public Sub New
```

Reference

DimensionCollection Class [▶ 1209]

DimensionCollection Overload [▶ 1211]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.1.2 DimensionCollection Constructor (Int32)

Initializes a new instance of an 1-Dimensional representing DimensionCollection [▶ 1209] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DimensionCollection(
    int length
)
```

VB

```
Public Sub New (
    length As Integer
)
```

Parameters

length	Type: System.Int32 The length.

Reference

DimensionCollection Class [▶ 1209]

DimensionCollection Overload [▶ 1211]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.1.3 DimensionCollection Constructor (.Int32.)

Initializes a new instance of the DimensionCollection [▶ 1209] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DimensionCollection(
    int[] dimLengths
)
```

VB

```
Public Sub New (
    dimLengths As Integer()
)
```

Parameters

dimLengths	Type: .System.Int32. The dim lengths.

Reference

DimensionCollection Class [▶ 1209]

DimensionCollection Overload [▶ 1211]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.1.4 DimensionCollection Constructor (IEnumerable.IDimension.)

Initializes a new instance of the DimensionCollection [▶ 1209] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DimensionCollection(
    IEnumerable<IDimension> coll
)
```

VB

```
Public Sub New (
    coll As IEnumerable(Of IDimension)
)
```

Parameters

coll	Type: System.Collections.Generic.IEnumerable.IDimension [▶ 1525]. The coll.

Reference

DimensionCollection Class [▶ 1209]

DimensionCollection Overload [▶ 1211]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.2 DimensionCollection Properties

The DimensionCollection [▶ 1209] type exposes the following members.

Properties

	Name	Description
	Count [▶ 1214]	Gets the number of elements contained in the ICollection.T..
	ElementCount [▶ 1215]	Gets the Number of elements in all Dimensions

	Name	Description
	IsReadOnly [▶ 1215]	Gets a value indicating whether the ICollection.T. is read-only.
	Item [▶ 1216]	Gets or sets the element at the specified index.
	LowerBounds [▶ 1217]	Gets the lower bounds.
	UpperBounds [▶ 1217]	Gets the upper bounds.

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.2.1 DimensionCollection.Count Property

Gets the number of elements contained in the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Count { get; }
```

VB

```
Public ReadOnly Property Count As Integer
    Get
```

Property Value

Type: Int32
The count.

Implements

ICollection.T..Count

Exceptions

Exception	Condition
NotImplementedException	

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.2.2 DimensionCollection.ElementCount Property

Gets the Number of elements in all Dimensions

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ElementCount { get; }
```

VB

```
Public ReadOnly Property ElementCount As Integer
    Get
```

Property Value

Type: Int32

Implements

IDimensionCollection.ElementCount [▶ 1529]

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.2.3 DimensionCollection.IsReadOnly Property

Gets a value indicating whether the ICollection.T. is read-only.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is read only; otherwise, false.

Implements

ICollection.T..IsReadOnly

Exceptions

Exception	Condition
NotImplementedException	

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.2.4 DimensionCollection.Item Property

Gets or sets the element at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDimension this[
    int index
] { get; set; }
```

VB

```
Public Default Property Item (
    index As Integer
) As IDimension
    Get
    Set
```

Parameters

index	Type: System.Int32 The index.

Return Value

Type: IDimension [▶ 1525]
IDimension.

Implements

IList.T..Item.Int32.

Exceptions

Exception	Condition
NotImplementedException	

Remarks

Index = 0 is the lowest dimesion.

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.2.5 DimensionCollection.LowerBounds Property

Gets the lower bounds.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int[] LowerBounds { get; }
```

VB

```
Public ReadOnly Property LowerBounds As Integer()
    Get
```

Property Value

Type: .Int32.
The lower bounds.

Implements

IDimensionCollection.LowerBounds [▶ 1529]

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.2.6 DimensionCollection.UpperBounds Property

Gets the upper bounds.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int[] UpperBounds { get; }
```

VB

```
Public ReadOnly Property UpperBounds As Integer()
    Get
```

Property Value

Type: .Int32.
The upper bounds.

Implements

IDimensionCollection.UpperBounds [▶ 1530]

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.3 DimensionCollection Methods

The DimensionCollection [▶ 1209] type exposes the following members.

Methods

	Name	Description
	Add [▶ 1219]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1219]	Returns a read only version of this DimensionCollection [▶ 1209].
	Clear [▶ 1220]	Removes all items from the ICollection.T..
	Contains [▶ 1221]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1221]	Copies to.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDimensionLengths [▶ 1222]	Gets an array the specifies the Lengths of each Array Dimension
	GetEnumerator [▶ 1223]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1223]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1224]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1225]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1226]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.3.1 DimensionCollection.Add Method

Adds an item to the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Add(
    IDimension item
)
```

VB

```
Public Sub Add (
    item As IDimension
)
```

Parameters

item	Type: TwinCAT.TypeSystem.IDimension [▶ 1525] The object to add to the ICollection.T..

Implements

ICollection.T..Add(T)

Exceptions

Exception	Condition
NotImplementedException	

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.3.2 DimensionCollection.AsReadOnly Method

Returns a read only version of this DimensionCollection [▶ 1209].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyDimensionCollection AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlyDimensionCollection
```

Field Value

Type: ReadOnlyDimensionCollection [▶ 1731]
As read only.

Return Value

Type: ReadOnlyDimensionCollection [▶ 1731]
ReadOnlyDimensionCollection.

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.3.3 DimensionCollection.Clear Method

Removes all items from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Clear()
```

VB

```
Public Sub Clear
```

Implements

ICollection.T..Clear.

Exceptions

Exception	Condition
NotImplementedException	

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.3.4 DimensionCollection.Contains Method

Determines whether the ICollection.T. contains a specific value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    IDimension item
)
```

VB

```
Public Function Contains (
    item As IDimension
) As Boolean
```

Parameters

item	Type: TwinCAT.TypeSystem.IDimension [▶ 1525] The object to locate in the ICollection.T..

Return Value

Type: Boolean
true if item is found in the ICollection.T.; otherwise, false.

Implements

ICollection.T..Contains(T)

Exceptions

Exception	Condition
NotImplementedException	

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.3.5 DimensionCollection.CopyTo Method

Copies to.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void CopyTo(
    IDimension[] array,
    int arrayIndex
)
```

VB

```
Public Sub CopyTo (
    array As IDimension(),
    arrayIndex As Integer
)
```

Parameters

array	Type: .TwinCAT.TypeSystem.IDimension [▶ 1525]. The array.
arrayIndex	Type: System.Int32 Index of the array.

Implements

ICollection.T..CopyTo(.T., Int32)

Exceptions

Exception	Condition
NotImplementedException	

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.3.6 DimensionCollection.GetDimensionLengths Method

Gets an array the specifies the Lengths of each Array Dimension

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int[] GetDimensionLengths()
```

VB

```
Public Function GetDimensionLengths As Integer()
```

Return Value

Type: .Int32.
System.Int32[].

Implements

IDimensionCollection.GetDimensionLengths. [▶ 1531]

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.3.7 DimensionCollection.GetEnumerator Method

Returns an enumerator that iterates through the collection.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumerator<IDimension> GetEnumerator()
```

VB

```
Public Function GetEnumerator As IEnumerator(Of IDimension)
```

Return Value

Type: IEnumerator.IDimension [▶ 1525].
A IEnumerator.T. that can be used to iterate through the collection.

Implements

IEnumerable.T..GetEnumerator.

Exceptions

Exception	Condition
NotImplementedException	

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.3.8 DimensionCollection.IndexOf Method

Determines the index of a specific item in the IList.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int IndexOf(
    IDimension item
)
```

VB

```
Public Function IndexOf (
    item As IDimension
) As Integer
```

Parameters

item	Type: TwinCAT.TypeSystem.IDimension [▶ 1525] The object to locate in the IList.T..

Return Value

Type: Int32
The index of item if found in the list; otherwise, -1.

Implements

IList.T..IndexOf(T)

Exceptions

Exception	Condition
NotImplementedException	

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.3.9 DimensionCollection.Insert Method

Inserts an item to the IList.T. at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Insert(
    int index,
    IDimension item
)
```

VB

```
Public Sub Insert (
    index As Integer,
    item As IDimension
)
```

Parameters

index	Type: System.Int32 The zero-based index at which item should be inserted.
item	Type: TwinCAT.TypeSystem.IDimension [▶ 1525] The object to insert into the IList.T..

Implements

IList.T..Insert(Int32, T)

Exceptions

Exception	Condition
NotImplementedException	

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.3.10 DimensionCollection.Remove Method

Removes the first occurrence of a specific object from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Remove(
    IDimension item
)
```

VB

```
Public Function Remove (
    item As IDimension
) As Boolean
```

Parameters

item	Type: TwinCAT.TypeSystem.IDimension [▶ 1525] The object to remove from the ICollection.T..

Return Value

Type: Boolean
true if item was successfully removed from the ICollection.T.; otherwise, false. This method also returns false if item is not found in the original ICollection.T..

Implements

ICollection.T..Remove(T)

Exceptions

Exception	Condition
NotImplementedException	

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.7.3.11 DimensionCollection.RemoveAt Method

Removes the IList.T. item at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void RemoveAt(
    int index
)
```

VB

```
Public Sub RemoveAt (
    index As Integer
)
```

Parameters

index	Type: System.Int32 The zero-based index of the item to remove.

Implements

IList.T..RemoveAt(Int32)

Exceptions

Exception	Condition
NotImplementedException	

Reference

DimensionCollection Class [▶ 1209]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.8 DynamicAliasInstance Class

Class DynamicAliasInstance. This class cannot be inherited.

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]
      TwinCAT.TypeSystem.DynamicAliasInstance

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class DynamicAliasInstance : DynamicSymbol,
    IAliasInstance, ISymbol, IAttributedInstance, IInstance, IBitSize
```

VB

```
Public NotInheritable Class DynamicAliasInstance
    Inherits DynamicSymbol
    Implements IAliasInstance, ISymbol, IAttributedInstance, IInstance,
    IBitSize
```

The DynamicAliasInstance type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1234]	Returns the enumeration of all dynamic member names. (Overrides DynamicSymbol.GetDynamicMemberNames. [▶ 1356].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex [▶ 1235]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Overrides DynamicObject.TryGetIndex(GetIndexBinder, .Object., Object.).)

	Name	Description
	TryGetMember [▶ 1236]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicSymbol.TryGetMember(GetMemberBinder, Object.) [▶ 1370].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex [▶ 1237]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Overrides DynamicObject.TrySetIndex(SetIndexBinder, .Object., Object).)
	TrySetMember [▶ 1237]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Overrides DynamicObject.TrySetMember(SetMemberBinder, Object).)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]

TwinCAT.TypeSystem.IAliasInstance [▶ 1490]

5.8.8.1 DynamicAliasInstance Properties

The DynamicAliasInstance [▶ 1227] type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicAliasInstance Class [▶ 1227]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.8.2 DynamicAliasInstance Methods

The DynamicAliasInstance [▶ 1227] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1234]	Returns the enumeration of all dynamic member names. (Overrides DynamicSymbol.GetDynamicMemberNames. [▶ 1356].)

	Name	Description
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex [▶ 1235]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Overrides DynamicObject.TryGetIndex(GetIndexBinder, .Object., Object.).)
	TryGetMember [▶ 1236]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicSymbol.TryGetMember(GetMemberBinder, Object.) [▶ 1370].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)

	Name	Description
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex [▶ 1237]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Overrides DynamicObject.TrySetIndex(SetIndexBinder, .Object., Object).)
	TrySetMember [▶ 1237]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Overrides DynamicObject.TrySetMember(SetMemberBinder, Object).)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicAliasInstance Class [▶ 1227]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.8.2.1 DynamicAliasInstance.GetDynamicMemberNames Method

Returns the enumeration of all dynamic member names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override IEnumerable<string> GetDynamicMemberNames()
```

VB

```
Public Overrides Function GetDynamicMemberNames As IEnumerable(Of String)
```

Return Value

Type: IEnumerable.String.
A sequence that contains dynamic member names.

Reference

DynamicAliasInstance Class [▶ 1227]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.8.2.2 DynamicAliasInstance.TryGetIndex Method

Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetIndex(
    GetIndexBinder binder,
    Object[] indexes,
    out Object result
)
```

VB

```
Public Overrides Function TryGetIndex (
    binder As GetIndexBinder,
    indexes As Object(),
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetIndexBinder Provides information about the operation.
indexes	Type: .System.Object. The indexes that are used in the operation. For example, for the sampleObject[3] operation in C# (sampleObject(3) in Visual Basic), where sampleObject is derived from the DynamicObject class, indexes[0] is equal to 3.
result	Type: System.Object. The result of the index operation.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicAliasInstance Class [▶ 1227]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.8.2.3 DynamicAliasInstance.TryGetMember Method

Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetMember(
    GetMemberBinder binder,
    out Object result
)
```

VB

```
Public Overrides Function TryGetMember (
    binder As GetMemberBinder,
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member on which the dynamic operation is performed. For example, for the Console.WriteLine(sampleObject.SampleProperty) statement, where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
result	Type: System.Object. The result of the get operation. For example, if the method is called for a property, you can assign the property value to result.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicAliasInstance Class [▶ 1227]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.8.2.4 DynamicAliasInstance.TrySetIndex Method

Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TrySetIndex(
    SetIndexBinder binder,
    Object[] indexes,
    Object value
)
```

VB

```
Public Overrides Function TrySetIndex (
    binder As SetIndexBinder,
    indexes As Object(),
    value As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.SetIndexBinder Provides information about the operation.
indexes	Type: .System.Object. The indexes that are used in the operation. For example, for the sampleObject[3] = 10 operation in C# (sampleObject(3) = 10 in Visual Basic), where sampleObject is derived from the DynamicObject class, indexes[][] is equal to 3.
value	Type: System.Object The value to set to the object that has the specified index. For example, for the sampleObject[3] = 10 operation in C# (sampleObject(3) = 10 in Visual Basic), where sampleObject is derived from the DynamicObject class, value is equal to 10.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.

Reference

DynamicAliasInstance Class [▶ 1227]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.8.2.5 DynamicAliasInstance.TrySetMember Method

Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TrySetMember(
    SetMemberBinder binder,
    Object value
)
```

VB

```
Public Overrides Function TrySetMember (
    binder As SetMemberBinder,
    value As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.SetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member to which the value is being assigned. For example, for the statement sampleObject.SampleProperty = "Test", where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
value	Type: System.Object The value to set to the member. For example, for sampleObject.SampleProperty = "Test", where sampleObject is an instance of the class derived from the DynamicObject class, the value is "Test".

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.)

Reference

DynamicAliasInstance Class [▶ 1227]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.8.3 DynamicAliasInstance Events

The DynamicAliasInstance [▶ 1227] type exposes the following members.

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicAliasInstance Class [▶ 1227]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.9 DynamicArrayInstance Class

Dynamic Array Instance

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]
      TwinCAT.TypeSystem.DynamicArrayInstance
        TwinCAT.TypeSystem.DynamicOversamplingArrayInstance [▶ 1255]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class DynamicArrayInstance : DynamicSymbol,
    IArrayInstance, ISymbol, IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Class DynamicArrayInstance
    Inherits DynamicSymbol
    Implements IArrayInstance, ISymbol, IAttributedInstance, IInstance,
    IBitSize
```

The DynamicArrayInstance type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	Dimensions [▶ 1245]	Gets the dimensions as read only collection.
	Elements [▶ 1245]	Gets the contained Array Elements as read only collection.

	Name	Description
	ElementType [▶ 1246]	Gets the type of the contained elements.
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	Item [▶ 1247]	Gets the ISymbol [▶ 1634] with the specified indices.
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1356]	Returns the enumeration of all dynamic member names. (Inherited from DynamicSymbol [▶ 1329].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnReadAnyValue [▶ 1357]	Handler function for reading ADS 'Any' Values. (Inherited from DynamicSymbol [▶ 1329].)
	OnReadRawValue [▶ 1358]	Handler function for reading Raw symbol value. (Inherited from DynamicSymbol [▶ 1329].)
	OnReadValue [▶ 1359]	Handler function for the (Inherited from DynamicSymbol [▶ 1329].)
	OnSetInstanceName [▶ 1359]	Sets a new InstanceName InstancePath (Inherited from DynamicSymbol [▶ 1329].)
	OnTryReadValue [▶ 1360]	Handler function for the (Inherited from DynamicSymbol [▶ 1329].)
	OnTryWriteValue [▶ 1361]	Handler Function for writing value. (Inherited from DynamicSymbol [▶ 1329].)
	OnWriteRawValue [▶ 1361]	Handler function for reading symbols raw value. (Inherited from DynamicSymbol [▶ 1329].)
	OnWriteValue [▶ 1362]	Handler Function for writing value. (Inherited from DynamicSymbol [▶ 1329].)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)

	Name	Description
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetElement(IList.. Int32.., ISymbol.) [▶ 1250]	Tries to get the array element with the specified indices (jagged array support).
	TryGetElement(.Int32., ISymbol.) [▶ 1251]	Tries to get the array element
	TryGetIndex [▶ 1252]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Overrides DynamicObject.TryGetIndex(GetIndexBinder, .Object., Object.).)
	TryGetMember [▶ 1370]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Inherited from DynamicSymbol [▶ 1329].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex [▶ 1253]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Overrides DynamicObject.TrySetIndex(SetIndexBinder, .Object., Object).)
	TrySetMember	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicObject.)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Fields

	Name	Description
	normalizedName [▶ 1388]	The normalized name of this . (Inherited from DynamicSymbol [▶ 1329].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.9.1 DynamicArrayInstance Properties

The DynamicArrayInstance [▶ 1239] type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	Dimensions [▶ 1245]	Gets the dimensions as read only collection.
	Elements [▶ 1245]	Gets the contained Array Elements as read only collection.
	ElementType [▶ 1246]	Gets the type of the contained elements.
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	Item [▶ 1247]	Gets the ISymbol [▶ 1634] with the specified indices.
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicArrayInstance Class [▶ 1239]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.9.1.1 DynamicArrayInstance.Dimensions Property

Gets the dimensions as read only collection.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyDimensionCollection Dimensions { get; }
```

VB

```
Public ReadOnly Property Dimensions As ReadOnlyDimensionCollection
    Get
```

Property Value

Type: ReadOnlyDimensionCollection [▶ 1731]
The dimensions.

Implements

IArrayInstance.Dimensions [▶ 1499]

Reference

DynamicArrayInstance Class [▶ 1239]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.9.1.2 DynamicArrayInstance.Elements Property

Gets the contained Array Elements as read only collection.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlySymbolCollection Elements { get; }
```

VB

```
Public ReadOnly Property Elements As ReadOnlySymbolCollection
    Get
```

Property Value

Type: ReadOnlySymbolCollection [▶ 1772]
The elements.

Implements

IArrayInstance.Elements [▶ 1500]

Reference

DynamicArrayInstance Class [▶ 1239]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.9.1.3 DynamicArrayInstance.ElementType Property

Gets the type of the contained elements.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType ElementType { get; }
```

VB

```
Public ReadOnly Property ElementType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the element.

Implements

IArrayInstance.ElementType [▶ 1500]

Reference

DynamicArrayInstance Class [▶ 1239]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.9.1.4 DynamicArrayInstance.Item Property

Gets the ISymbol [▶ 1634] with the specified indices.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ISymbol this[
    int[] indices
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    indices As Integer()
) As ISymbol
    Get
```

Parameters

indices	Type: .System.Int32. The indices.

Return Value

Type: ISymbol [▶ 1634]
ISymbol.

Implements

IArrayInstance.Item..Int32.. [▶ 1501]

Exceptions

Exception	Condition
ArgumentOutOfRangeException	indices

Reference

DynamicArrayInstance Class [▶ 1239]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.9.2 DynamicArrayInstance Methods

The DynamicArrayInstance [▶ 1239] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1356]	Returns the enumeration of all dynamic member names. (Inherited from DynamicSymbol [▶ 1329].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnReadAnyValue [▶ 1357]	Handler function for reading ADS 'Any' Values. (Inherited from DynamicSymbol [▶ 1329].)
	OnReadRawValue [▶ 1358]	Handler function for reading Raw symbol value. (Inherited from DynamicSymbol [▶ 1329].)
	OnReadValue [▶ 1359]	Handler function for the (Inherited from DynamicSymbol [▶ 1329].)
	OnSetInstanceName [▶ 1359]	Sets a new InstanceName InstancePath (Inherited from DynamicSymbol [▶ 1329].)
	OnTryReadValue [▶ 1360]	Handler function for the (Inherited from DynamicSymbol [▶ 1329].)
	OnTryWriteValue [▶ 1361]	Handler Function for writing value. (Inherited from DynamicSymbol [▶ 1329].)
	OnWriteRawValue [▶ 1361]	Handler function for reading symbols raw value. (Inherited from DynamicSymbol [▶ 1329].)
	OnWriteValue [▶ 1362]	Handler Function for writing value. (Inherited from DynamicSymbol [▶ 1329].)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)

	Name	Description
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetElement(IList.. Int32.., ISymbol.) [▶ 1250]	Tries to get the array element with the specified indices (jagged array support).
	TryGetElement(.Int32., ISymbol.) [▶ 1251]	Tries to get the array element
	TryGetIndex [▶ 1252]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Overrides DynamicObject.TryGetIndex(GetIndexBinder, .Object., Object.).)
	TryGetMember [▶ 1370]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Inherited from DynamicSymbol [▶ 1329].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex [▶ 1253]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Overrides DynamicObject.TrySetIndex(SetIndexBinder, .Object., Object).)
	TrySetMember	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicObject.)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicArrayInstance Class [▶ 1239]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.9.2.1 DynamicArrayInstance.TryGetElement Method

Overload List

	Name	Description
	TryGetElement(IList.. Int32.., ISymbol.) [▶ 1250]	Tries to get the array element with the specified indices (jagged array support).
	TryGetElement(.Int32., ISymbol.) [▶ 1251]	Tries to get the array element

Reference

DynamicArrayInstance Class [▶ 1239]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicArrayInstance.TryGetElement Method (IList..Int32.., ISymbol.)

Tries to get the array element with the specified indices (jagged array support).

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetElement(
    IList<int[]> jaggedIndices,
    out ISymbol symbol
)
```

VB

```
Public Function TryGetElement (
    jaggedIndices As IList(Of Integer()),
    <OutAttribute> ByRef symbol As ISymbol
) As Boolean
```

Parameters

jaggedIndices	Type: System.Collections.Generic.IList..Int32.. The jagged indices list.
symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634]. The symbol.

Return Value

Type: Boolean
true if found, false if the jagged indices specifiers is out-of-range.

Implements

IArrayInstance.TryGetElement(IList..Int32.., ISymbol.) [▶ 1502]

Exceptions

Exception	Condition
ArgumentNullException	jaggedIndices
ArgumentOutOfRangeException	jaggedIndices

Reference

DynamicArrayInstance Class [▶ 1239]

TryGetElement Overload [▶ 1250]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicArrayInstance.TryGetElement Method (.Int32., ISymbol.)

Tries to get the array element

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetElement(
    int[] indices,
    out ISymbol symbol
)
```

VB

```
Public Function TryGetElement (
    indices As Integer(),
    <OutAttribute> ByRef symbol As ISymbol
) As Boolean
```

Parameters

indices	Type: .System.Int32. The indices.
symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634]. The found Array element symbol (out-parameter).

Return Value

Type: Boolean
true if found, false if the indices specifiers is out-of-range.

Implements

IArrayInstance.TryGetElement(.Int32., ISymbol.) [▶ 1503]

Reference

DynamicArrayInstance Class [▶ 1239]

TryGetElement Overload [▶ 1250]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.9.2.2 DynamicArrayInstance.TryGetIndex Method

Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetIndex(
    GetIndexBinder binder,
    Object[] indexes,
    out Object result
)
```

VB

```
Public Overrides Function TryGetIndex (
    binder As GetIndexBinder,
    indexes As Object(),
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetIndexBinder Provides information about the operation.
indexes	Type: .System.Object. The indexes that are used in the operation. For example, for the sampleObject[3] operation in C# (sampleObject(3) in Visual Basic), where sampleObject is derived from the DynamicObject class, indexes[0] is equal to 3.
result	Type: System.Object. The result of the index operation.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicArrayInstance Class [▶ 1239]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.9.2.3 DynamicArrayInstance.TrySetIndex Method

Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TrySetIndex(
    SetIndexBinder binder,
    Object[] indexes,
    Object value
)
```

VB

```
Public Overrides Function TrySetIndex (
    binder As SetIndexBinder,
    indexes As Object(),
    value As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.SetIndexBinder Provides information about the operation.

indexes	Type: .System.Object. The indexes that are used in the operation. For example, for the sampleObject[3] = 10 operation in C# (sampleObject(3) = 10 in Visual Basic), where sampleObject is derived from the DynamicObject class, indexes[][] is equal to 3.
value	Type: System.Object The value to set to the object that has the specified index. For example, for the sampleObject[3] = 10 operation in C# (sampleObject(3) = 10 in Visual Basic), where sampleObject is derived from the DynamicObject class, value is equal to 10.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.

Reference

DynamicArrayInstance Class [▶ 1239]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.9.3 DynamicArrayInstance Events

The DynamicArrayInstance [▶ 1239] type exposes the following members.

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicArrayInstance Class [▶ 1239]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.9.4 DynamicArrayInstance Fields

The DynamicArrayInstance [▶ 1239] type exposes the following members.

Fields

	Name	Description
	normalizedName [▶ 1388]	The normalized name of this . (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicArrayInstance Class [▶ 1239]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.10 DynamicOversamplingArrayInstance Class

Dynamic Array Instance

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]
      TwinCAT.TypeSystem.DynamicArrayInstance [▶ 1239]
        TwinCAT.TypeSystem.DynamicOversamplingArrayInstance

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class DynamicOversamplingArrayInstance : DynamicArrayInstance,
    IOversamplingArrayInstance, IArrayInstance, ISymbol, IAttributedInstance, IInstance,
    IBitSize
```

VB

```
Public NotInheritable Class DynamicOversamplingArrayInstance
    Inherits DynamicArrayInstance
    Implements IOversamplingArrayInstance, IArrayInstance, ISymbol, IAttributedInstance,
    IInstance, IBitSize
```

The DynamicOversamplingArrayInstance type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	Dimensions [▶ 1245]	Gets the dimensions as read only collection. (Inherited from DynamicArrayInstance [▶ 1239].)
	Elements [▶ 1245]	Gets the contained Array Elements as read only collection. (Inherited from DynamicArrayInstance [▶ 1239].)

	Name	Description
	ElementType [▶ 1246]	Gets the type of the contained elements. (Inherited from DynamicArrayInstance [▶ 1239].)
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	Item [▶ 1247]	Gets the ISymbol [▶ 1634] with the specified indices. (Inherited from DynamicArrayInstance [▶ 1239].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	OversamplingElement [▶ 1261]	Gets the oversampling element.
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1356]	Returns the enumeration of all dynamic member names. (Inherited from DynamicSymbol [▶ 1329].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetElement(IList..Int32.., ISymbol.) [▶ 1250]	Tries to get the array element with the specified indices (jagged array support). (Inherited from DynamicArrayInstance [▶ 1239].)

	Name	Description
	TryGetElement(.Int32., ISymbol.) [▶ 1251]	Tries to get the array element (Inherited from DynamicArrayInstance [▶ 1239].)
	TryGetIndex [▶ 1252]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicArrayInstance [▶ 1239].)
	TryGetMember [▶ 1370]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Inherited from DynamicSymbol [▶ 1329].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex [▶ 1253]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicArrayInstance [▶ 1239].)
	TrySetMember	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicObject.)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.10.1 DynamicOversamplingArrayInstance Properties

The DynamicOversamplingArrayInstance [▶ 1255] type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	Dimensions [▶ 1245]	Gets the dimensions as read only collection. (Inherited from DynamicArrayInstance [▶ 1239].)
	Elements [▶ 1245]	Gets the contained Array Elements as read only collection. (Inherited from DynamicArrayInstance [▶ 1239].)
	ElementType [▶ 1246]	Gets the type of the contained elements. (Inherited from DynamicArrayInstance [▶ 1239].)
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	Item [▶ 1247]	Gets the ISymbol [▶ 1634] with the specified indices. (Inherited from DynamicArrayInstance [▶ 1239].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	OversamplingElement [▶ 1261]	Gets the oversampling element.
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicOversamplingArrayInstance Class [▶ 1255]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.10.1.1 DynamicOversamplingArrayInstance.OversamplingElement Property

Gets the oversampling element.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ISymbol OversamplingElement { get; }
```

VB

```
Public ReadOnly Property OversamplingElement As ISymbol
    Get
```

Property Value

Type: ISymbol [▶ 1634]
The oversampling element.

Implements

IOversamplingArrayInstance.OversamplingElement [▶ 1570]

Exceptions

Exception	Condition
NotImplementedException	

Reference

DynamicOversamplingArrayInstance Class [▶ 1255]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.10.2 DynamicOversamplingArrayInstance Methods

The DynamicOversamplingArrayInstance [▶ 1255] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1356]	Returns the enumeration of all dynamic member names. (Inherited from DynamicSymbol [▶ 1329].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This

	Name	Description
		method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetElement(IList.., Int32.., ISymbol.) [▶ 1250]	Tries to get the array element with the specified indices (jagged array support). (Inherited from DynamicArrayInstance [▶ 1239].)
	TryGetElement(.Int32., ISymbol.) [▶ 1251]	Tries to get the array element (Inherited from DynamicArrayInstance [▶ 1239].)
	TryGetIndex [▶ 1252]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicArrayInstance [▶ 1239].)
	TryGetMember [▶ 1370]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex [▶ 1253]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicArrayInstance [▶ 1239].)
	TrySetMember	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicObject.)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicOversamplingArrayInstance Class [▶ 1255]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.10.3 DynamicOversamplingArrayInstance Events

The DynamicOversamplingArrayInstance [▶ 1255] type exposes the following members.

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicOversamplingArrayInstance Class [▶ 1255]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.11 DynamicPointerInstance Class

Dynamic Pointer Instance

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]
      TwinCAT.TypeSystem.DynamicPointerInstance
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class DynamicPointerInstance : DynamicSymbol,
    IPointerInstance, ISymbol, IAttributedInstance, IInstance, IBitSize
```

VB

```
Public NotInheritable Class DynamicPointerInstance
    Inherits DynamicSymbol
    Implements IPointerInstance, ISymbol, IAttributedInstance, IInstance,
    IBitSize
```

The DynamicPointerInstance type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Reference [▶ 1269]	Gets the resolved reference of Pointer / Reference
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1272]	Returns the enumeration of all dynamic member names. (Overrides DynamicSymbol.GetDynamicMemberNames. [▶ 1356].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)

	Name	Description
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetMember [▶ 1272]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicSymbol.TryGetMember(GetMemberBinder, Object.) [▶ 1370].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicObject.)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.11.1 DynamicPointerInstance Properties

The DynamicPointerInstance [▶ 1264] type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Reference [▶ 1269]	Gets the resolved reference of Pointer / Reference
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicPointerInstance Class [▶ 1264]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.11.1.1 DynamicPointerInstance.Reference Property

Gets the resolved reference of Pointer / Reference

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ISymbol Reference { get; }
```

VB

```
Public ReadOnly Property Reference As ISymbol
    Get
```

Property Value

Type: ISymbol [▶ 1634]
The reference.

Implements

IPointerInstance.Reference [▶ 1574]

Reference

DynamicPointerInstance Class [▶ 1264]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.11.2 DynamicPointerInstance Methods

The DynamicPointerInstance [▶ 1264] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1272]	Returns the enumeration of all dynamic member names. (Overrides DynamicSymbol.GetDynamicMemberNames. [▶ 1356].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetMember [▶ 1272]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicSymbol.TryGetMember(GetMemberBinder, Object.) [▶ 1370].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicObject.)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicPointerInstance Class [▶ 1264]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.11.2.1 DynamicPointerInstance.GetDynamicMemberNames Method

Returns the enumeration of all dynamic member names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override IEnumerable<string> GetDynamicMemberNames()
```

VB

```
Public Overrides Function GetDynamicMemberNames As IEnumerable(Of String)
```

Return Value

Type: IEnumerable.String.
A sequence that contains dynamic member names.

Reference

DynamicPointerInstance Class [▶ 1264]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.11.2.2 DynamicPointerInstance.TryGetMember Method

Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetMember(
    GetMemberBinder binder,
    out Object result
)
```

VB

```
Public Overrides Function TryGetMember (
    binder As GetMemberBinder,
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member on which the dynamic operation is performed. For example, for the Console.WriteLine(sampleObject.SampleProperty) statement, where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
result	Type: System.Object. The result of the get operation. For example, if the method is called for a property, you can assign the property value to result.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicPointerInstance Class [▶ 1264]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.11.3 DynamicPointerInstance Events

The DynamicPointerInstance [▶ 1264] type exposes the following members.

Events

	Name	Description
⚡	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
⚡	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicPointerInstance Class [▶ 1264]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.12 DynamicPointerValue Class

Class DynamicPointerValue.

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicValue [▶ 1407]
      TwinCAT.TypeSystem.DynamicPointerValue
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class DynamicPointerValue : DynamicValue
```

VB

```
Public Class DynamicPointerValue
    Inherits DynamicValue
```

The DynamicPointerValue type exposes the following members.

Properties

	Name	Description
	Age [▶ 1411]	Gets the age of the value (last successful read of the value) (Inherited from DynamicValue [▶ 1407].)
	CachedRaw [▶ 1412]	Gets the cached Raw internal Data. (Inherited from DynamicValue [▶ 1407].)
	DataType [▶ 1412]	Gets the data type bound to this IValue [▶ 1659] (Inherited from DynamicValue [▶ 1407].)
	IsPrimitive [▶ 1413]	Gets a value indicating whether this IValue [▶ 1659] is a primitive value. (Inherited from DynamicValue [▶ 1407].)
	ResolvedType [▶ 1413]	Gets the resolved type. (Inherited from DynamicValue [▶ 1407].)
	Symbol [▶ 1414]	Gets the symbol that is bound to this value. (Inherited from DynamicValue [▶ 1407].)
	UtcTimeStamp [▶ 1414]	Gets the Time stamp of the last successful read of the Value. (Inherited from DynamicValue [▶ 1407].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1279]	Returns the enumeration of all dynamic member names. (Overrides DynamicValue.GetDynamicMemberNames. [▶ 1417].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)

	Name	Description
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Read [▶ 1418]	Reads the value (via ADS) (Inherited from DynamicValue [▶ 1407].)
	ReadMember [▶ 1418]	Reads the specified member element. (Inherited from DynamicValue [▶ 1407].)
	ResolveValue [▶ 1419]	Resolves the Value object to its primitive value. (Inherited from DynamicValue [▶ 1407].)
	ToString [▶ 1419]	Returns a String that represents this instance. (Inherited from DynamicValue [▶ 1407].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert [▶ 1420]	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicValue [▶ 1407].)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetArrayElement Values [▶ 1421]	Returns Array Element values. (Inherited from DynamicValue [▶ 1407].)
	TryGetIndex [▶ 1421]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicValue [▶ 1407].)
	TryGetIndexValue(.Int32., Object.) [▶ 1423]	Reads the specified array element. (Inherited from DynamicValue [▶ 1407].)
	TryGetIndexValue(.Object., Object.) [▶ 1423]	Tries the get index value. (Inherited from DynamicValue [▶ 1407].)
	TryGetMember [▶ 1424]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Inherited from DynamicValue [▶ 1407].)

	Name	Description
	TryGetMemberValue [▶ 1280]	Tries the get member value. (Overrides DynamicValue.TryGetMemberValue(String, Object.) [▶ 1425].)
	TryInvoke [▶ 1426]	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicValue [▶ 1407].)
	TryInvokeMember [▶ 1427]	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicValue [▶ 1407].)
	TryResolveValue [▶ 1428]	Tries to resolves the Value object to its primitive value. (Inherited from DynamicValue [▶ 1407].)
	TrySetIndex [▶ 1428]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicValue [▶ 1407].)
	TrySetIndexValue [▶ 1429]	Tries to set the indexed value on Arrays (Inherited from DynamicValue [▶ 1407].)
	TrySetMember [▶ 1430]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicValue [▶ 1407].)
	TrySetMemberValue [▶ 1431]	Tries to Set a Member/Property Value (Inherited from DynamicValue [▶ 1407].)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	Write [▶ 1432]	Writes the value (via ADS) (Inherited from DynamicValue [▶ 1407].)
	WriteMember [▶ 1432]	Writes the specified member element. (Inherited from DynamicValue [▶ 1407].)

Fields

	Name	Description
	_symbol [▶ 1433]	Symbol that is bound to this value. (Inherited from DynamicValue [▶ 1407].)
	s_pointerDeref [▶ 1281]	Pointer Deref indicator
	valueFactory [▶ 1434]	The value factory (Inherited from DynamicValue [▶ 1407].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.DynamicValue [▶ 1407]

5.8.12.1 DynamicPointerValue Properties

The DynamicPointerValue [▶ 1274] type exposes the following members.

Properties

	Name	Description
	Age [▶ 1411]	Gets the age of the value (last successful read of the value) (Inherited from DynamicValue [▶ 1407].)
	CachedRaw [▶ 1412]	Gets the cached Raw internal Data. (Inherited from DynamicValue [▶ 1407].)
	DataType [▶ 1412]	Gets the data type bound to this IValue [▶ 1659] (Inherited from DynamicValue [▶ 1407].)
	IsPrimitive [▶ 1413]	Gets a value indicating whether this IValue [▶ 1659] is a primitive value. (Inherited from DynamicValue [▶ 1407].)
	ResolvedType [▶ 1413]	Gets the resolved type. (Inherited from DynamicValue [▶ 1407].)
	Symbol [▶ 1414]	Gets the symbol that is bound to this value. (Inherited from DynamicValue [▶ 1407].)
	UtcTimeStamp [▶ 1414]	Gets the Time stamp of the last successful read of the Value. (Inherited from DynamicValue [▶ 1407].)

Reference

DynamicPointerValue Class [▶ 1274]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.12.2 DynamicPointerValue Methods

The DynamicPointerValue [▶ 1274] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1279]	Returns the enumeration of all dynamic member names. (Overrides DynamicValue.GetDynamicMemberNames. [▶ 1417].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)

	Name	Description
	Read [▶ 1418]	Reads the value (via ADS) (Inherited from DynamicValue [▶ 1407].)
	ReadMember [▶ 1418]	Reads the specified member element. (Inherited from DynamicValue [▶ 1407].)
	ResolveValue [▶ 1419]	Resolves the Value object to its primitive value. (Inherited from DynamicValue [▶ 1407].)
	ToString [▶ 1419]	Returns a String that represents this instance. (Inherited from DynamicValue [▶ 1407].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert [▶ 1420]	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicValue [▶ 1407].)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetArrayElementValues [▶ 1421]	Returns Array Element values. (Inherited from DynamicValue [▶ 1407].)
	TryGetIndex [▶ 1421]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicValue [▶ 1407].)
	TryGetIndexValue(.Int32., Object.) [▶ 1423]	Reads the specified array element. (Inherited from DynamicValue [▶ 1407].)
	TryGetIndexValue(.Object., Object.) [▶ 1423]	Tries the get index value. (Inherited from DynamicValue [▶ 1407].)
	TryGetMember [▶ 1424]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Inherited from DynamicValue [▶ 1407].)
	TryGetMemberValue [▶ 1280]	Tries the get member value. (Overrides DynamicValue.TryGetMemberValue(String, Object.) [▶ 1425].)
	TryInvoke [▶ 1426]	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicValue [▶ 1407].)

	Name	Description
	TryInvokeMember [▶ 1427]	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicValue [▶ 1407].)
	TryResolveValue [▶ 1428]	Tries to resolves the Value object to its primitive value. (Inherited from DynamicValue [▶ 1407].)
	TrySetIndex [▶ 1428]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicValue [▶ 1407].)
	TrySetIndexValue [▶ 1429]	Tries to set the indexed value on Arrays (Inherited from DynamicValue [▶ 1407].)
	TrySetMember [▶ 1430]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicValue [▶ 1407].)
	TrySetMemberValue [▶ 1431]	Tries to Set a Member/Property Value (Inherited from DynamicValue [▶ 1407].)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	Write [▶ 1432]	Writes the value (via ADS) (Inherited from DynamicValue [▶ 1407].)
	WriteMember [▶ 1432]	Writes the specified member element. (Inherited from DynamicValue [▶ 1407].)

Reference

DynamicPointerValue Class [▶ 1274]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.12.2.1 DynamicPointerValue.GetDynamicMemberNames Method

Returns the enumeration of all dynamic member names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override IEnumerable<string> GetDynamicMemberNames()
```

VB

```
Public Overrides Function GetDynamicMemberNames As IEnumerable(Of String)
```

Return Value

Type: IEnumerable.String.
A sequence that contains dynamic member names.

Reference

DynamicPointerValue Class [▶ 1274]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.12.2.2 DynamicPointerValue.TryGetMemberValue Method

Tries the get member value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetMemberValue(
    string name,
    out Object result
)
```

VB

```
Public Overrides Function TryGetMemberValue (
    name As String,
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

name	Type: System.String The name.
result	Type: System.Object. The result.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Implements

IStructValue.TryGetMemberValue(String, Object.) [▶ 1626]

Exceptions

Exception	Condition
SymbolException [▶ 606]	

Reference

DynamicPointerValue Class [▶ 1274]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.12.3 DynamicPointerValue Fields

The DynamicPointerValue [▶ 1274] type exposes the following members.

Fields

	Name	Description
	_symbol [▶ 1433]	Symbol that is bound to this value. (Inherited from DynamicValue [▶ 1407].)
	s_pointerDeref [▶ 1281]	Pointer Deref indicator
	valueFactory [▶ 1434]	The value factory (Inherited from DynamicValue [▶ 1407].)

Reference

DynamicPointerValue Class [▶ 1274]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.12.3.1 DynamicPointerValue.s_pointerDeref Field

Pointer Deref indicator

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static string s_pointerDeref
```

VB

```
Public Shared s_pointerDeref As String
```

Field Value

Type: String

Reference

DynamicPointerValue Class [▶ 1274]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.13 DynamicReferenceInstance Class

Dynamic Reference Instance

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]
      TwinCAT.TypeSystem.DynamicReferenceInstance
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class DynamicReferenceInstance : DynamicSymbol,
    IReferenceInstance, ISymbol, IAttributedInstance, IInstance, IBitSize
```

VB

```
Public NotInheritable Class DynamicReferenceInstance
    Inherits DynamicSymbol
    Implements IReferenceInstance, ISymbol, IAttributedInstance, IInstance,
    IBitSize
```

The DynamicReferenceInstance type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1289]	Returns the enumeration of all dynamic member names. (Overrides DynamicSymbol.GetDynamicMemberNames. [▶ 1356].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex [▶ 1289]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Overrides DynamicObject.TryGetIndex(GetIndexBinder, .Object., Object.).)
	TryGetMember [▶ 1290]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicSymbol.TryGetMember(GetMemberBinder, Object.) [▶ 1370].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex [▶ 1291]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Overrides DynamicObject.TrySetIndex(SetIndexBinder, .Object., Object).)
	TrySetMember [▶ 1292]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Overrides DynamicObject.TrySetMember(SetMemberBinder, Object).)

	Name	Description
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.13.1 DynamicReferenceInstance Properties

The DynamicReferenceInstance [▶ 1281] type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicReferenceInstance Class [▶ 1281]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.13.2 DynamicReferenceInstance Methods

The DynamicReferenceInstance [▶ 1281] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1289]	Returns the enumeration of all dynamic member names. (Overrides DynamicSymbol.GetDynamicMemberNames. [▶ 1356].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)

	Name	Description
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex [▶ 1289]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Overrides DynamicObject.TryGetIndex(GetIndexBinder, .Object., Object.).)
	TryGetMember [▶ 1290]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicSymbol.TryGetMember(GetMemberBinder, Object.) [▶ 1370].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex [▶ 1291]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Overrides DynamicObject.TrySetIndex(SetIndexBinder, .Object., Object).)
	TrySetMember [▶ 1292]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Overrides DynamicObject.TrySetMember(SetMemberBinder, Object).)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicReferenceInstance Class [▶ 1281]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.13.2.1 DynamicReferenceInstance.GetDynamicMemberNames Method

Returns the enumeration of all dynamic member names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override IEnumerable<string> GetDynamicMemberNames()
```

VB

```
Public Overrides Function GetDynamicMemberNames As IEnumerable(Of String)
```

Return Value

Type: IEnumerable.String.
A sequence that contains dynamic member names.

Reference

DynamicReferenceInstance Class [▶ 1281]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.13.2.2 DynamicReferenceInstance.TryGetIndex Method

Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetIndex(
    GetIndexBinder binder,
    Object[] indexes,
    out Object result
)
```

VB

```
Public Overrides Function TryGetIndex (
    binder As GetIndexBinder,
    indexes As Object(),
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetIndexBinder Provides information about the operation.
indexes	Type: .System.Object. The indexes that are used in the operation. For example, for the sampleObject[3] operation in C# (sampleObject(3) in Visual Basic), where sampleObject is derived from the DynamicObject class, indexes[0] is equal to 3.
result	Type: System.Object. The result of the index operation.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicReferenceInstance Class [▶ 1281]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.13.2.3 DynamicReferenceInstance.TryGetMember Method

Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetMember(
    GetMemberBinder binder,
    out Object result
)
```

VB

```
Public Overrides Function TryGetMember (
    binder As GetMemberBinder,
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member on which the dynamic operation is performed. For example, for the Console.WriteLine(sampleObject.SampleProperty) statement, where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
result	Type: System.Object. The result of the get operation. For example, if the method is called for a property, you can assign the property value to result.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicReferenceInstance Class [▶ 1281]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.13.2.4 DynamicReferenceInstance.TrySetIndex Method

Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TrySetIndex(
    SetIndexBinder binder,
    Object[] indexes,
    Object value
)
```

VB

```
Public Overrides Function TrySetIndex (
    binder As SetIndexBinder,
    indexes As Object(),
    value As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.SetIndexBinder Provides information about the operation.
indexes	Type: .System.Object. The indexes that are used in the operation. For example, for the sampleObject[3] = 10 operation in C# (sampleObject(3) = 10 in Visual Basic), where sampleObject is derived from the DynamicObject class, indexes[][] is equal to 3.
value	Type: System.Object The value to set to the object that has the specified index. For example, for the sampleObject[3] = 10 operation in C# (sampleObject(3) = 10 in Visual Basic), where sampleObject is derived from the DynamicObject class, value is equal to 10.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.

Reference

DynamicReferenceInstance Class [▶ 1281]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.13.2.5 DynamicReferenceInstance.TrySetMember Method

Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TrySetMember(
    SetMemberBinder binder,
    Object value
)
```

VB

```
Public Overrides Function TrySetMember (
    binder As SetMemberBinder,
    value As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.SetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member to which the value is being assigned. For example, for the statement sampleObject.SampleProperty = "Test", where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.

value	Type: System.Object The value to set to the member. For example, for sampleObject.SampleProperty = "Test", where sampleObject is an instance of the class derived from the DynamicObject class, the value is "Test".

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.)

Reference

DynamicReferenceInstance Class [▶ 1281]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.13.3 DynamicReferenceInstance Events

The DynamicReferenceInstance [▶ 1281] type exposes the following members.

Events

	Name	Description
⚡	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
⚡	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicReferenceInstance Class [▶ 1281]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.14 DynamicReferenceValue Class

Class DynamicReferenceValue.

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicValue [▶ 1407]
      TwinCAT.TypeSystem.DynamicReferenceValue
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class DynamicReferenceValue : DynamicValue
```

VB

```
Public Class DynamicReferenceValue
    Inherits DynamicValue
```

The DynamicReferenceValue type exposes the following members.

Properties

	Name	Description
	Age [▶ 1411]	Gets the age of the value (last successful read of the value) (Inherited from DynamicValue [▶ 1407].)
	CachedRaw [▶ 1412]	Gets the cached Raw internal Data. (Inherited from DynamicValue [▶ 1407].)
	DataType [▶ 1412]	Gets the data type bound to this IValue [▶ 1659] (Inherited from DynamicValue [▶ 1407].)
	IsPrimitive [▶ 1413]	Gets a value indicating whether this IValue [▶ 1659] is a primitive value. (Inherited from DynamicValue [▶ 1407].)
	ResolvedType [▶ 1413]	Gets the resolved type. (Inherited from DynamicValue [▶ 1407].)
	Symbol [▶ 1414]	Gets the symbol that is bound to this value. (Inherited from DynamicValue [▶ 1407].)
	UtcTimeStamp [▶ 1414]	Gets the Time stamp of the last successful read of the Value. (Inherited from DynamicValue [▶ 1407].)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1417]	Returns the enumeration of all dynamic member names. (Inherited from DynamicValue [▶ 1407].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Read [▶ 1418]	Reads the value (via ADS) (Inherited from DynamicValue [▶ 1407].)
	ReadMember [▶ 1299]	Reads the specified member element. (Overrides DynamicValue.ReadMember(ISymbol) [▶ 1418].)
	ResolveValue [▶ 1419]	Resolves the Value object to its primitive value. (Inherited from DynamicValue [▶ 1407].)
	ToString [▶ 1419]	Returns a String that represents this instance. (Inherited from DynamicValue [▶ 1407].)

	Name	Description
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert [▶ 1420]	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicValue [▶ 1407].)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetArrayElement Values [▶ 1421]	Returns Array Element values. (Inherited from DynamicValue [▶ 1407].)
	TryGetIndex [▶ 1421]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicValue [▶ 1407].)
	TryGetIndexValue(.Int32., Object.) [▶ 1423]	Reads the specified array element. (Inherited from DynamicValue [▶ 1407].)
	TryGetIndexValue(. Object., Object.) [▶ 1423]	Tries the get index value. (Inherited from DynamicValue [▶ 1407].)
	TryGetMember [▶ 1424]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Inherited from DynamicValue [▶ 1407].)
	TryGetMemberValue [▶ 1425]	Tries the get member value. (Inherited from DynamicValue [▶ 1407].)
	TryInvoke [▶ 1426]	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicValue [▶ 1407].)
	TryInvokeMember [▶ 1427]	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicValue [▶ 1407].)
	TryResolveValue [▶ 1428]	Tries to resolves the Value object to its primitive value. (Inherited from DynamicValue [▶ 1407].)
	TrySetIndex [▶ 1428]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicValue [▶ 1407].)

	Name	Description
	TrySetIndexValue [▶ 1429]	Tries to set the indexed value on Arrays (Inherited from DynamicValue [▶ 1407].)
	TrySetMember [▶ 1430]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicValue [▶ 1407].)
	TrySetMemberValue [▶ 1431]	Tries to Set a Member/Property Value (Inherited from DynamicValue [▶ 1407].)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	Write [▶ 1432]	Writes the value (via ADS) (Inherited from DynamicValue [▶ 1407].)
	WriteMember [▶ 1432]	Writes the specified member element. (Inherited from DynamicValue [▶ 1407].)

Fields

	Name	Description
	_symbol [▶ 1433]	Symbol that is bound to this value. (Inherited from DynamicValue [▶ 1407].)
	valueFactory [▶ 1434]	The value factory (Inherited from DynamicValue [▶ 1407].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.DynamicValue [▶ 1407]

5.8.14.1 DynamicReferenceValue Properties

The DynamicReferenceValue [▶ 1293] type exposes the following members.

Properties

	Name	Description
	Age [▶ 1411]	Gets the age of the value (last successful read of the value) (Inherited from DynamicValue [▶ 1407].)
	CachedRaw [▶ 1412]	Gets the cached Raw internal Data. (Inherited from DynamicValue [▶ 1407].)
	DataType [▶ 1412]	Gets the data type bound to this IValue [▶ 1659] (Inherited from DynamicValue [▶ 1407].)
	IsPrimitive [▶ 1413]	Gets a value indicating whether this IValue [▶ 1659] is a primitive value. (Inherited from DynamicValue [▶ 1407].)
	ResolvedType [▶ 1413]	Gets the resolved type. (Inherited from DynamicValue [▶ 1407].)
	Symbol [▶ 1414]	Gets the symbol that is bound to this value. (Inherited from DynamicValue [▶ 1407].)
	UtcTimeStamp [▶ 1414]	Gets the Time stamp of the last successful read of the Value. (Inherited from DynamicValue [▶ 1407].)

Reference

DynamicReferenceValue Class [▶ 1293]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.14.2 DynamicReferenceValue Methods

The DynamicReferenceValue [▶ 1293] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1417]	Returns the enumeration of all dynamic member names. (Inherited from DynamicValue [▶ 1407].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Read [▶ 1418]	Reads the value (via ADS) (Inherited from DynamicValue [▶ 1407].)
	ReadMember [▶ 1299]	Reads the specified member element. (Overrides DynamicValue.ReadMember(ISymbol) [▶ 1418].)
	ResolveValue [▶ 1419]	Resolves the Value object to its primitive value. (Inherited from DynamicValue [▶ 1407].)
	ToString [▶ 1419]	Returns a String that represents this instance. (Inherited from DynamicValue [▶ 1407].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert [▶ 1420]	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicValue [▶ 1407].)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)

	Name	Description
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetArrayElementValues [▶ 1421]	Returns Array Element values. (Inherited from DynamicValue [▶ 1407].)
	TryGetIndex [▶ 1421]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicValue [▶ 1407].)
	TryGetIndexValue(.Int32., Object.) [▶ 1423]	Reads the specified array element. (Inherited from DynamicValue [▶ 1407].)
	TryGetIndexValue(.Object., Object.) [▶ 1423]	Tries the get index value. (Inherited from DynamicValue [▶ 1407].)
	TryGetMember [▶ 1424]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Inherited from DynamicValue [▶ 1407].)
	TryGetMemberValue [▶ 1425]	Tries the get member value. (Inherited from DynamicValue [▶ 1407].)
	TryInvoke [▶ 1426]	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicValue [▶ 1407].)
	TryInvokeMember [▶ 1427]	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicValue [▶ 1407].)
	TryResolveValue [▶ 1428]	Tries to resolves the Value object to its primitive value. (Inherited from DynamicValue [▶ 1407].)
	TrySetIndex [▶ 1428]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicValue [▶ 1407].)
	TrySetIndexValue [▶ 1429]	Tries to set the indexed value on Arrays (Inherited from DynamicValue [▶ 1407].)
	TrySetMember [▶ 1430]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicValue [▶ 1407].)
	TrySetMemberValue [▶ 1431]	Tries to Set a Member/Property Value (Inherited from DynamicValue [▶ 1407].)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	Write [▶ 1432]	Writes the value (via ADS) (Inherited from DynamicValue [▶ 1407].)

	Name	Description
	WriteMember [▶ 1432]	Writes the specified member element. (Inherited from DynamicValue [▶ 1407].)

Reference

DynamicReferenceValue Class [▶ 1293]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.14.2.1 DynamicReferenceValue.ReadMember Method

Reads the specified member element.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected override Object ReadMember(
    ISymbol memberInstance
)
```

VB

```
Protected Overrides Function ReadMember (
    memberInstance As ISymbol
) As Object
```

Parameters

memberInstance	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The member instance.

Return Value

Type: Object

Reference

DynamicReferenceValue Class [▶ 1293]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.14.3 DynamicReferenceValue Fields

The DynamicReferenceValue [▶ 1293] type exposes the following members.

Fields

	Name	Description
	_symbol [▶ 1433]	Symbol that is bound to this value. (Inherited from DynamicValue [▶ 1407].)
	valueFactory [▶ 1434]	The value factory (Inherited from DynamicValue [▶ 1407].)

Reference

DynamicReferenceValue Class [▶ 1293]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.15 DynamicRpcStructInstance Class

Dynamic struct instance with RPC Methods.

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]
      TwinCAT.TypeSystem.DynamicStructInstance [▶ 1315]
        TwinCAT.TypeSystem.DynamicRpcStructInstance

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class DynamicRpcStructInstance : DynamicStructInstance,
    IRpcStructInstance, IStructInstance, ISymbol, IAttributedInstance, IInstance,
    IBitSize, IRpcCallableInstance
```

VB

```
Public NotInheritable Class DynamicRpcStructInstance
    Inherits DynamicStructInstance
    Implements IRpcStructInstance, IStructInstance, ISymbol, IAttributedInstance,
    IInstance, IBitSize, IRpcCallableInstance
```

The DynamicRpcStructInstance type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	HasRpcMethods [▶ 1322]	Gets a value indicating whether this instance has RPC methods (Inherited from DynamicStructInstance [▶ 1315].)
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	MemberInstances [▶ 1322]	Gets the member instances of the Struct Instance [▶ 1615]. (Inherited from DynamicStructInstance [▶ 1315].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	RpcMethods [▶ 1306]	Gets the Method descriptions for the IRpcCallableType [▶ 1596]
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1309]	Gets the dynamic member names. (Overrides DynamicStructInstance.GetDynamicMemberNames. [▶ 1326].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	InvokeRpcMethod [▶ 1309]	Invokes the specified method.
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)

	Name	Description
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetMember [▶ 1310]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicStructInstance.TryGetMember(GetMemberBinder, Object.) [▶ 1326].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember [▶ 1311]	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Overrides DynamicObject.TryInvokeMember(InvokeMemberBinder, .Object., Object.).)
	TryInvokeRpcMethod(String, .Object., Object.) [▶ 1312]	Tries to invoke the specified method.
	TryInvokeRpcMethod(IRpcMethod, .Object., Object.) [▶ 1313]	Tries to invoke the specified method.
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember [▶ 1314]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Overrides DynamicStructInstance.TrySetMember(SetMemberBinder, Object) [▶ 1327].)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]

TwinCAT.TypeSystem.IStructInstance [▶ 1615]

TwinCAT.TypeSystem.IRpcStructInstance [▶ 1604]

TwinCAT.TypeSystem.IRpcCallableInstance [▶ 1591]

5.8.15.1 DynamicRpcStructInstance Properties

The DynamicRpcStructInstance [▶ 1300] type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	HasRpcMethods [▶ 1322]	Gets a value indicating whether this instance has RPC methods (Inherited from DynamicStructInstance [▶ 1315].)
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	MemberInstances [▶ 1322]	Gets the member instances of the Struct Instance [▶ 1615]. (Inherited from DynamicStructInstance [▶ 1315].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	RpcMethods [▶ 1306]	Gets the Method descriptions for the IRpcCallableType [▶ 1596]
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicRpcStructInstance Class [▶ 1300]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.15.1.1 DynamicRpcStructInstance.RpcMethods Property

Gets the Method descriptions for the IRpcCallableType [▶ 1596]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyRpcMethodCollection RpcMethods { get; }
```

VB

```
Public ReadOnly Property RpcMethods As ReadOnlyRpcMethodCollection
    Get
```

Property Value

Type: ReadOnlyRpcMethodCollection [▶ 1762]
The methods.

Implements

IRpcCallableInstance.RpcMethods [▶ 1592]

Reference

DynamicRpcStructInstance Class [▶ 1300]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.15.2 DynamicRpcStructInstance Methods

The DynamicRpcStructInstance [▶ 1300] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1309]	Gets the dynamic member names. (Overrides DynamicStructInstance.GetDynamicMemberNames. [▶ 1326].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	InvokeRpcMethod [▶ 1309]	Invokes the specified method.
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetMember [▶ 1310]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicStructInstance.TryGetMember(GetMemberBinder, Object.) [▶ 1326].)

	Name	Description
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember [▶ 1311]	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Overrides DynamicObject.TryInvokeMember(InvokeMemberBinder, .Object., Object.).)
	TryInvokeRpcMethod(String, .Object., Object.) [▶ 1312]	Tries to invoke the specified method.
	TryInvokeRpcMethod(IRpcMethod, .Object., Object.) [▶ 1313]	Tries to invoke the specified method.
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember [▶ 1314]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Overrides DynamicStructInstance.TrySetMember(SetMemberBinder, Object) [▶ 1327].)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicRpcStructInstance Class [▶ 1300]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.15.2.1 DynamicRpcStructInstance.GetDynamicMemberNames Method

Gets the dynamic member names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override IEnumerable<string> GetDynamicMemberNames()
```

VB

```
Public Overrides Function GetDynamicMemberNames As IEnumerable(Of String)
```

Return Value

Type: IEnumerable.String.
IEnumerable<System.String>.

Reference

DynamicRpcStructInstance Class [▶ 1300]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.15.2.2 DynamicRpcStructInstance.InvokeRpcMethod Method

Invokes the specified method.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object InvokeRpcMethod(
    string methodName,
    Object[] parameters
)
```

VB

```
Public Function InvokeRpcMethod (
    methodName As String,
    parameters As Object()
) As Object
```

Parameters

methodName	Type: System.String Name of the method.

parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
The return value of the RPC Method

Implements

IRpcCallableInstance.InvokeRpcMethod(String, .Object.) [▶ 1593]

Remarks

To indicate a PLC Method for remote ads access, the attribute 'TcRpcEnable' must be declared on the method declaration (see example).

Examples

RPC Method definition and implementation

```
(* Declaration *)
{attribute 'TcRpcEnable'}
METHOD RpcMethod1 : INT
VAR_INPUT
i1 : INT;
END_VAR
(* Implementation *)
RpcMethod1 := i1 + 1;
```

Reference

DynamicRpcStructInstance Class [▶ 1300]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.15.2.3 DynamicRpcStructInstance.TryGetMember Method

Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetMember(
    GetMemberBinder binder,
    out Object result
)
```

VB

```
Public Overrides Function TryGetMember (
    binder As GetMemberBinder,
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member on which the dynamic operation is performed. For example, for the Console.WriteLine(sampleObject.SampleProperty) statement, where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
result	Type: System.Object. The result of the get operation. For example, if the method is called for a property, you can assign the property value to result.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicRpcStructInstance Class [▶ 1300]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.15.2.4 DynamicRpcStructInstance.TryInvokeMember Method

Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryInvokeMember(
    InvokeMemberBinder binder,
    Object[] args,
    out Object result
)
```

VB

```
Public Overrides Function TryInvokeMember (
    binder As InvokeMemberBinder,
    args As Object(),
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.InvokeMemberBinder Provides information about the dynamic operation. The binder.Name property provides the name of the member on which the dynamic operation is performed. For example, for the statement sampleObject.SampleMethod(100), where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleMethod". The binder.IgnoreCase property specifies whether the member name is case-sensitive.

args	Type: .System.Object. The arguments that are passed to the object member during the invoke operation. For example, for the statement sampleObject.SampleMethod(100), where sampleObject is derived from the DynamicObject class, args[][] is equal to 100.
result	Type: System.Object. The result of the member invocation.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.)

Reference

DynamicRpcStructInstance Class [▶ 1300]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.15.2.5 DynamicRpcStructInstance.TryInvokeRpcMethod Method

Overload List

	Name	Description
	TryInvokeRpcMethod(String, .Object., Object.) [▶ 1312]	Tries to invoke the specified method.
	TryInvokeRpcMethod(IRpcMethod, .Object., Object.) [▶ 1313]	Tries to invoke the specified method.

Reference

DynamicRpcStructInstance Class [▶ 1300]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicRpcStructInstance.TryInvokeRpcMethod Method (String, .Object., Object.)

Tries to invoke the specified method.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int TryInvokeRpcMethod(
    string methodName,
    Object[] args,
    out Object result
)
```

VB

```
Public Function TryInvokeRpcMethod (
    methodName As String,
    args As Object(),
    <OutAttribute> ByRef result As Object
) As Integer
```

Parameters

methodName	Type: System.String Name of the method.
args	Type: .System.Object. The arguments.
result	Type: System.Object. The result.

Return Value

Type: Int32
true if the call succeeds, false otherwise.

Implements

IRpcCallableInstance.TryInvokeRpcMethod(String, .Object., Object.) [▶ 1594]

Exceptions

Exception	Condition
NotImplementedException	

Reference

DynamicRpcStructInstance Class [▶ 1300]

TryInvokeRpcMethod Overload [▶ 1312]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicRpcStructInstance.TryInvokeRpcMethod Method (IRpcMethod, .Object., Object.)

Tries to invoke the specified method.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int TryInvokeRpcMethod(
    IRpcMethod method,
    Object[] args,
    out Object result
)
```

VB

```
Public Function TryInvokeRpcMethod (
    method As IRpcMethod,
    args As Object(),
    <OutAttribute> ByRef result As Object
) As Integer
```

Parameters

method	Type: TwinCAT.TypeSystem.IRpcMethod [▶ 1597] The method.
args	Type: .System.Object. The arguments.
result	Type: System.Object. The result.

Return Value

Type: Int32
true if the call succeeds, false otherwise.

Implements

IRpcCallableInstance.TryInvokeRpcMethod(IRpcMethod, .Object., Object.) [▶ 1595]

Exceptions

Exception	Condition
NotImplementedException	

Reference

DynamicRpcStructInstance Class [▶ 1300]

TryInvokeRpcMethod Overload [▶ 1312]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.15.2.6 DynamicRpcStructInstance.TrySetMember Method

Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TrySetMember(
    SetMemberBinder binder,
    Object value
)
```

VB

```
Public Overrides Function TrySetMember (
    binder As SetMemberBinder,
    value As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.SetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member to which the value is being assigned. For example, for the statement sampleObject.SampleProperty = "Test", where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
value	Type: System.Object The value to set to the member. For example, for sampleObject.SampleProperty = "Test", where sampleObject is an instance of the class derived from the DynamicObject class, the value is "Test".

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.)

Reference

DynamicRpcStructInstance Class [▶ 1300]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.15.3 DynamicRpcStructInstance Events

The DynamicRpcStructInstance [▶ 1300] type exposes the following members.

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicRpcStructInstance Class [▶ 1300]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.16 DynamicStructInstance Class

Dynamic struct instance

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]
      TwinCAT.TypeSystem.DynamicStructInstance
        TwinCAT.TypeSystem.DynamicRpcStructInstance [▶ 1300]
        TwinCAT.TypeSystem.DynamicVirtualStructInstance [▶ 1434]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class DynamicStructInstance : DynamicSymbol,
    IStructInstance, ISymbol, IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Class DynamicStructInstance
    Inherits DynamicSymbol
    Implements IStructInstance, ISymbol, IAttributedInstance, IInstance,
    IBitSize
```

The DynamicStructInstance type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	HasRpcMethods [▶ 1322]	Gets a value indicating whether this instance has RPC methods
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	MemberInstances [▶ 1322]	Gets the member instances of the Struct Instance [▶ 1615].
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1326]	Returns the enumeration of all dynamic member names. (Overrides DynamicSymbol.GetDynamicMemberNames. [▶ 1356].)

	Name	Description
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnReadAnyValue [▶ 1357]	Handler function for reading ADS 'Any' Values. (Inherited from DynamicSymbol [▶ 1329].)
	OnReadRawValue [▶ 1358]	Handler function for reading Raw symbol value. (Inherited from DynamicSymbol [▶ 1329].)
	OnReadValue [▶ 1359]	Handler function for the (Inherited from DynamicSymbol [▶ 1329].)
	OnSetInstanceName [▶ 1359]	Sets a new InstanceName InstancePath (Inherited from DynamicSymbol [▶ 1329].)
	OnTryReadValue [▶ 1360]	Handler function for the (Inherited from DynamicSymbol [▶ 1329].)
	OnTryWriteValue [▶ 1361]	Handler Function for writing value. (Inherited from DynamicSymbol [▶ 1329].)
	OnWriteRawValue [▶ 1361]	Handler function for reading symbols raw value. (Inherited from DynamicSymbol [▶ 1329].)
	OnWriteValue [▶ 1362]	Handler Function for writing value. (Inherited from DynamicSymbol [▶ 1329].)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)

	Name	Description
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetMember [▶ 1326]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicSymbol.TryGetMember(GetMemberBinder, Object.) [▶ 1370].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember [▶ 1327]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Overrides DynamicObject.TrySetMember(SetMemberBinder, Object).)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Fields

	Name	Description
	normalizedDict [▶ 1328]	Dictionary of normalized Instance Names
	normalizedName [▶ 1388]	The normalized name of this . (Inherited from DynamicSymbol [▶ 1329].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.16.1 DynamicStructInstance Properties

The DynamicStructInstance [▶ 1315] type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	HasRpcMethods [▶ 1322]	Gets a value indicating whether this instance has RPC methods
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	MemberInstances [▶ 1322]	Gets the member instances of the Struct Instance [▶ 1615].
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicStructInstance Class [▶ 1315]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.16.1.1 DynamicStructInstance.HasRpcMethods Property

Gets a value indicating whether this instance has RPC methods

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool HasRpcMethods { get; }
```

VB

```
Public ReadOnly Property HasRpcMethods As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has RPC methods; otherwise, false.

Implements

IStructInstance.HasRpcMethods [▶ 1618]

Remarks

If the struct instance supports RPC Methods, then the instance class is also supporting IRpcStructInstance [▶ 1604]

Reference

DynamicStructInstance Class [▶ 1315]

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IRpcStructInstance [▶ 1604]

TwinCAT.TypeSystem.IRpcMethod [▶ 1597]

TwinCAT.TypeSystem.IRpcMethodParameter [▶ 1601]

5.8.16.1.2 DynamicStructInstance.MemberInstances Property

Gets the member instances of the Struct Instance [▶ 1615].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlySymbolCollection MemberInstances { get; }
```

VB

```
Public ReadOnly Property MemberInstances As ReadOnlySymbolCollection
    Get
```

Property Value

Type: ReadOnlySymbolCollection [▶ 1772]
The member instances.

Implements

IStructInstance.MemberInstances [▶ 1619]

Reference

DynamicStructInstance Class [▶ 1315]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.16.2 DynamicStructInstance Methods

The DynamicStructInstance [▶ 1315] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1326]	Returns the enumeration of all dynamic member names. (Overrides DynamicSymbol.GetDynamicMemberNames. [▶ 1356].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)

	Name	Description
	OnReadAnyValue [▶ 1357]	Handler function for reading ADS 'Any' Values. (Inherited from DynamicSymbol [▶ 1329].)
	OnReadRawValue [▶ 1358]	Handler function for reading Raw symbol value. (Inherited from DynamicSymbol [▶ 1329].)
	OnReadValue [▶ 1359]	Handler function for the (Inherited from DynamicSymbol [▶ 1329].)
	OnSetInstanceName [▶ 1359]	Sets a new InstanceName InstancePath (Inherited from DynamicSymbol [▶ 1329].)
	OnTryReadValue [▶ 1360]	Handler function for the (Inherited from DynamicSymbol [▶ 1329].)
	OnTryWriteValue [▶ 1361]	Handler Function for writing value. (Inherited from DynamicSymbol [▶ 1329].)
	OnWriteRawValue [▶ 1361]	Handler function for reading symbols raw value. (Inherited from DynamicSymbol [▶ 1329].)
	OnWriteValue [▶ 1362]	Handler Function for writing value. (Inherited from DynamicSymbol [▶ 1329].)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)

	Name	Description
	TryGetMember [▶ 1326]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicSymbol.TryGetMember(GetMemberBinder, Object.) [▶ 1370].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember [▶ 1327]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Overrides DynamicObject.TrySetMember(SetMemberBinder, Object).)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicStructInstance Class [▶ 1315]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.16.2.1 DynamicStructInstance.GetDynamicMemberNames Method

Returns the enumeration of all dynamic member names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override IEnumerable<string> GetDynamicMemberNames()
```

VB

```
Public Overrides Function GetDynamicMemberNames As IEnumerable(Of String)
```

Return Value

Type: IEnumerable.String.
A sequence that contains dynamic member names.

Reference

DynamicStructInstance Class [▶ 1315]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.16.2.2 DynamicStructInstance.TryGetMember Method

Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetMember(
    GetMemberBinder binder,
    out Object result
)
```

VB

```
Public Overrides Function TryGetMember (
    binder As GetMemberBinder,
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member on which the dynamic operation is performed. For example, for the Console.WriteLine(sampleObject.SampleProperty) statement, where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.

result	Type: System.Object. The result of the get operation. For example, if the method is called for a property, you can assign the property value to result.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicStructInstance Class [▶ 1315]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.16.2.3 DynamicStructInstance.TrySetMember Method

Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TrySetMember(
    SetMemberBinder binder,
    Object value
)
```

VB

```
Public Overrides Function TrySetMember (
    binder As SetMemberBinder,
    value As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.SetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member to which the value is being assigned. For example, for the statement sampleObject.SampleProperty = "Test", where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
value	Type: System.Object The value to set to the member. For example, for sampleObject.SampleProperty = "Test", where sampleObject is an instance of the class derived from the DynamicObject class, the value is "Test".

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.)

Reference

DynamicStructInstance Class [▶ 1315]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.16.3 DynamicStructInstance Events

The DynamicStructInstance [▶ 1315] type exposes the following members.

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicStructInstance Class [▶ 1315]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.16.4 DynamicStructInstance Fields

The DynamicStructInstance [▶ 1315] type exposes the following members.

Fields

	Name	Description
	normalizedDict [▶ 1328]	Dictionary of normalized Instance Names
	normalizedName [▶ 1388]	The normalized name of this . (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicStructInstance Class [▶ 1315]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.16.4.1 DynamicStructInstance.normalizedDict Field

Dictionary of normalized Instance Names

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected Dictionary<string, ISymbol> normalizedDict
```

VB

```
Protected normalizedDict As Dictionary(Of String, ISymbol)
```

Field Value

Type: Dictionary.String, ISymbol [▶ 1634].

Reference

DynamicStructInstance Class [▶ 1315]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17 DynamicSymbol Class

Dynamic Symbol [▶ 1634] object.

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicSymbol
      More... [▶ 1335]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class DynamicSymbol : DynamicObject,
    IDynamicSymbol, ISymbol, IAttributedInstance, IInstance, IBitSize,
    IValueSymbol3, IValueSymbol2, IValueSymbol, IValueRawSymbol
```

VB

```
Public Class DynamicSymbol
    Inherits DynamicObject
    Implements IDynamicSymbol, ISymbol, IAttributedInstance, IInstance,
    IBitSize, IValueSymbol3, IValueSymbol2, IValueSymbol, IValueRawSymbol
```

The DynamicSymbol type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol
	AccessRights [▶ 1337]	Gets the access rights.
	Attributes [▶ 1337]	Gets the Symbol Attributes
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits.
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes
	Category [▶ 1339]	Gets the category.
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556]

	Name	Description
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol
	ContextMask [▶ 1341]	Gets the context mask.
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556].
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations).
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.))
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type.
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent.
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type.
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only.
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive.
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static.
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters),
	NotificationSettings [▶ 1350]	Gets the notification settings.
	Parent [▶ 1350]	Gets the parent Symbol
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes.
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634]
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556].

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	GetDynamicMemberNames [▶ 1356]	Returns the enumeration of all dynamic member names. (Overrides DynamicObject.GetDynamicMemberNames..)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnReadAnyValue [▶ 1357]	Handler function for reading ADS 'Any' Values.
	OnReadRawValue [▶ 1358]	Handler function for reading Raw symbol value.
	OnReadValue [▶ 1359]	Handler function for the
	OnSetInstanceName [▶ 1359]	Sets a new InstanceName InstancePath
	OnTryReadValue [▶ 1360]	Handler function for the
	OnTryWriteValue [▶ 1361]	Handler Function for writing value.
	OnWriteRawValue [▶ 1361]	Handler function for reading symbols raw value.
	OnWriteValue [▶ 1362]	Handler Function for writing value.
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol.
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol.
	ToString [▶ 1369]	Returns a String that represents this instance. (Overrides Object.ToString..)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)

	Name	Description
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetMember [▶ 1370]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicObject.TryGetMember(GetMemberBinder, Object.).)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683]
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicObject.)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol.
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value.
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683]
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value

	Name	Description
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol.
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol.

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed.
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed.

Operators

	Name	Description
	Equality [▶ 1386]	Operator==
	Inequality [▶ 1387]	Implements the != operator.

Fields

	Name	Description
	normalizedName [▶ 1388]	The normalized name of this .

Extension Methods

	Name	Description
	PollValuesAnnotated(IObservable.Unit.) [▶ 882]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence (Defined by ValueSymbolExtensions [▶ 877].)
	PollValuesAnnotated(TimeSpan) [▶ 883]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time. (Defined by ValueSymbolExtensions [▶ 877].)
	WhenValueChanged [▶ 884]	Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object.) [▶ 888]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)

	Name	Description
	WriteValues(IObservable.Object., Action.Exception.) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., CancellationToken) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception., CancellationToken) [▶ 891]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)

Remarks

The DynamicSymbol adds dynamic run time behaviour to the ISymbol [▶ 1634]/IValueSymbol [▶ 1683]. That means e.g. for StructSymbols that .NET Properties are defined and dispatched at runtime to the structs fields like they are defined in TwinCAT / ADS Types. Indexed access to Array Symbols is another example where the dynamic runtime support takes place.

Examples

Sample for the dynamic resolution of Symbols:

Dynamic Symbol access

```
/// <summary>
/// Defines the entry point of the application.
/// </summary>
/// <param name="args">The arguments.</param>
static void Main(string[] args)
{
    // Get the AdsAddress from command-line arguments
    AmsAddress address = ArgParser.Parse(args);

    using (TcAdsClient client = new TcAdsClient())
    {
    // Connect to the target device
    client.Connect(address);

    // Usage of "dynamic" Type and Symbols (>= .NET4 only)
    SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.DynamicTree);
    IAdsSymbolLoader dynLoader = (IAdsSymbolLoader)SymbolLoaderFactory.Create(client, settings);
```

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IDynamicSymbol [▶ 1531]

TwinCAT.TypeSystem.IValueSymbol [▶ 1683]

TwinCAT.TypeSystem.ISymbol [▶ 1634]

System.Dynamic.DynamicObject

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicSymbol
      TwinCAT.TypeSystem.DynamicAliasInstance [▶ 1227]
      TwinCAT.TypeSystem.DynamicArrayInstance [▶ 1239]
      TwinCAT.TypeSystem.DynamicPointerInstance [▶ 1264]
      TwinCAT.TypeSystem.DynamicReferenceInstance [▶ 1281]
      TwinCAT.TypeSystem.DynamicStructInstance [▶ 1315]
      TwinCAT.TypeSystem.DynamicUnionInstance [▶ 1396]

5.8.17.1 DynamicSymbol Properties

The DynamicSymbol [▶ 1329] type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329]
	AccessRights [▶ 1337]	Gets the access rights.
	Attributes [▶ 1337]	Gets the Symbol Attributes
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits.
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes
	Category [▶ 1339]	Gets the category.
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556]
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329]
	ContextMask [▶ 1341]	Gets the context mask.
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556].
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations).
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.))
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type.
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent.

	Name	Description
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type.
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only.
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive.
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static.
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters),
	NotificationSettings [▶ 1350]	Gets the notification settings.
	Parent [▶ 1350]	Gets the parent Symbol
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes.
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634]
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556].

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.1 DynamicSymbol._InnerSymbol Property

Inner symbol object wrapped by this DynamicSymbol [▶ 1329]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IValueSymbol _InnerSymbol { get; }
```

VB

```
Public ReadOnly Property _InnerSymbol As IValueSymbol
    Get
```

Property Value

Type: IValueSymbol [▶ 1683]
The inner symbol.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.2 DynamicSymbol.AccessRights Property

Gets the access rights.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolAccessRights AccessRights { get; }
```

VB

```
Public ReadOnly Property AccessRights As SymbolAccessRights
    Get
```

Property Value

Type: SymbolAccessRights [▶ 1818]
The access rights.

Implements

IValueSymbol.AccessRights [▶ 1687]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.3 DynamicSymbol.Attributes Property

Gets the Symbol Attributes

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyTypeAttributeCollection Attributes { get; }
```

VB

```
Public ReadOnly Property Attributes As ReadOnlyTypeAttributeCollection
    Get
```

Property Value

Type: ReadOnlyTypeAttributeCollection [▶ 1776]
The attributes.

Implements

IAttributedInstance.Attributes [▶ 1513]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.4 DynamicSymbol.BitSize Property

Gets the size of the IDataType [▶ 1517] in bits.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int BitSize { get; }
```

VB

```
Public ReadOnly Property BitSize As Integer
    Get
```

Property Value

Type: Int32
The size of the bit.

Implements

IBitSize.BitSize [▶ 1515]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.5 DynamicSymbol.ByteSize Property

Gets the (aligned) size of of the Type/Instance in Bytes

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ByteSize { get; }
```

VB

```
Public ReadOnly Property ByteSize As Integer
    Get
```

Property Value

Type: Int32
The size of the byte.

Implements

IBitSize.ByteSize [▶ 1516]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.6 DynamicSymbol.Category Property

Gets the category.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeCategory Category { get; }
```

VB

```
Public ReadOnly Property Category As DataTypeCategory
    Get
```

Property Value

Type: DataTypeCategory [▶ 1189]
The category.

Implements

ISymbol.Category [▶ 1636]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.7 DynamicSymbol.Comment Property

Gets the comment of the IInstance [▶ 1556]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Comment { get; }
```

VB

```
Public ReadOnly Property Comment As String
    Get
```

Property Value

Type: String
The comment.

Implements

IInstance.Comment [▶ 1558]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.8 DynamicSymbol.Connection Property

Gets the connection bound to this DynamicSymbol [▶ 1329]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IConnection Connection { get; }
```

VB

```
Public ReadOnly Property Connection As IConnection
    Get
```

Property Value

Type: IConnection [▶ 46]
The connection.

Implements

IValueSymbol2.Connection [▶ 1698]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.9 DynamicSymbol.ContextMask Property

Gets the context mask.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public byte ContextMask { get; }
```

VB

```
Public ReadOnly Property ContextMask As Byte
    Get
```

Property Value

Type: Byte
The context mask.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.10 DynamicSymbol.DataType Property

Gets the IDataType [▶ 1517] of the IInstance [▶ 1556].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType DataType { get; }
```

VB

```
Public ReadOnly Property DataType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the data.

Implements

IInstance.DataType [▶ 1559]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.11 DynamicSymbol.HasValue Property

Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations).

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool HasValue { get; }
```

VB

```
Public ReadOnly Property HasValue As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has value; otherwise, false.

Implements

IValueRawSymbol.HasValue [▶ 1678]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.12 DynamicSymbol.InstanceName Property

Gets the name of the instance (without periods (.)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string InstanceName { get; }
```

VB

```
Public ReadOnly Property InstanceName As String
    Get
```

Property Value

Type: String
The name of the instance.

Implements

IInstance.InstanceName [▶ 1559]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.13 DynamicSymbol.InstancePath Property

Gets the relative / absolute access path to the instance (with periods (.))

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string InstancePath { get; }
```

VB

```
Public ReadOnly Property InstancePath As String
    Get
```

Property Value

Type: String
The instance path.

Implements

IInstance.InstancePath [▶ 1560]

Remarks

If this path is relative or absolute depends on the context. IMember [▶ 1562] are using relative paths, ISymbol [▶ 1634]s are using absolute ones.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.14 DynamicSymbol.IsBitType Property

Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsBitType { get; }
```

VB

```
Public ReadOnly Property IsBitType As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is bit mapping; otherwise, false.

Implements

IBitSize.IsBitType [▶ 1516]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.15 DynamicSymbol.IsByteAligned Property

Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsByteAligned { get; }
```

VB

```
Public ReadOnly Property IsByteAligned As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is byte aligned; otherwise, false.

Implements

IBitSize.IsByteAligned [▶ 1517]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.16 DynamicSymbol.IsContainerType Property

Gets a value indicating whether this Symbol is acontainer type.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsContainerType { get; }
```

VB

```
Public ReadOnly Property IsContainerType As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is container type; otherwise, false.

Implements

ISymbol.IsContainerType [▶ 1637]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.17 DynamicSymbol.IsPersistent Property

Gets a value indicating whether this ISymbol [▶ 1634] is persistent.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsPersistent { get; }
```

VB

```
Public ReadOnly Property IsPersistent As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is persistent; otherwise, false.

Implements

ISymbol.IsPersistent [▶ 1637]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.18 DynamicSymbol.IsPointer Property

Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsPointer { get; }
```

VB

```
Public ReadOnly Property IsPointer As Boolean
    Get
```

Property Value

Type: Boolean
true if is ReferenceTo, otherwise false.

Implements

IInstance.IsPointer [▶ 1560]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.19 DynamicSymbol.IsPrimitiveType Property

Gets a value indicating whether this instance is a primitive type.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsPrimitiveType { get; }
```

VB

```
Public ReadOnly Property IsPrimitiveType As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is primitive type; otherwise, false.

Implements

ISymbol.IsPrimitiveType [▶ 1638]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.20 DynamicSymbol.IsReadOnly Property

Gets a value indicating whether this ISymbol [▶ 1634] is read only.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is read only; otherwise, false.

Implements

ISymbol.IsReadOnly [▶ 1638]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.21 DynamicSymbol.IsRecursive Property

Gets a value indicating whether this instance is recursive.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsRecursive { get; }
```

VB

```
Public ReadOnly Property IsRecursive As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is recursive; otherwise, false.

Implements

ISymbol.IsRecursive [▶ 1639]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.22 DynamicSymbol.IsReference Property

Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReference { get; }
```

VB

```
Public ReadOnly Property IsReference As Boolean
    Get
```

Property Value

Type: Boolean
true if is ReferenceTo, otherwise false.

Implements

IInstance.IsReference [▶ 1561]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.23 DynamicSymbol.IsStatic Property

Gets a value indicating whether this instance is static.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsStatic { get; }
```

VB

```
Public ReadOnly Property IsStatic As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is static; otherwise, false.

Implements

IInstance.IsStatic [▶ 1561]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.24 DynamicSymbol.NormalizedName Property

Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters),

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string NormalizedName { get; }
```

VB

```
Public ReadOnly Property NormalizedName As String
    Get
```

Property Value

Type: String
The normalized instance name (can be the same like InstanceName [▶ 1559]

Implements

IDynamicSymbol.NormalizedName [▶ 1534]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

IInstance.InstanceName [▶ 1559]

5.8.17.1.25 DynamicSymbol.NotificationSettings Property

Gets the notification settings.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public INotificationSettings NotificationSettings { get; set; }
```

VB

```
Public Property NotificationSettings As INotificationSettings
    Get
    Set
```

Property Value

Type: INotificationSettings [▶ 1566]
The notification settings.

Implements

IValueSymbol.NotificationSettings [▶ 1688]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.26 DynamicSymbol.Parent Property

Gets the parent Symbol

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ISymbol Parent { get; }
```

VB

```
Public ReadOnly Property Parent As ISymbol
    Get
```

Property Value

Type: ISymbol [▶ 1634]
The parent.

Implements

ISymbol.Parent [▶ 1639]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.27 DynamicSymbol.Size Property

Gets the size of the IInstance [▶ 1556] in bytes.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Size { get; }
```

VB

```
Public ReadOnly Property Size As Integer
    Get
```

Property Value

Type: Int32
The size of the IInstance [▶ 1556] in bytes.

Implements

IBitSize.Size [▶ 1517]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.28 DynamicSymbol.SubSymbols Property

Gets the SubSymbols of the ISymbol [▶ 1634]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlySymbolCollection SubSymbols { get; }
```

VB

```
Public ReadOnly Property SubSymbols As ReadOnlySymbolCollection
    Get
```

Property Value

Type: ReadOnlySymbolCollection [▶ 1772]

Implements

ISymbol.SubSymbols [▶ 1640]

Remarks

Only used for Array and Struct instances. Otherwise empty

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.1.29 DynamicSymbol.TypeName Property

Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string TypeName { get; }
```

VB

```
Public ReadOnly Property TypeName As String
    Get
```

Property Value

Type: String
The name of the type.

Implements

IInstance.TypeName [▶ 1562]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2 DynamicSymbol Methods

The DynamicSymbol [▶ 1329] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1356]	Returns the enumeration of all dynamic member names. (Overrides DynamicObject.GetDynamicMemberNames.)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Overrides Object.GetHashCode.)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	OnReadAnyValue [▶ 1357]	Handler function for reading ADS 'Any' Values.
	OnReadRawValue [▶ 1358]	Handler function for reading Raw symbol value.
	OnReadValue [▶ 1359]	Handler function for the
	OnSetInstanceName [▶ 1359]	Sets a new InstanceName InstancePath
	OnTryReadValue [▶ 1360]	Handler function for the
	OnTryWriteValue [▶ 1361]	Handler Function for writing value.
	OnWriteRawValue [▶ 1361]	Handler function for reading symbols raw value.
	OnWriteValue [▶ 1362]	Handler Function for writing value.
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329].

	Name	Description
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329].
	ToString [▶ 1369]	Returns a String that represents this instance. (Overrides Object.ToString..)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetMember [▶ 1370]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicObject.TryGetMember(GetMemberBinder, Object.).)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683]
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicObject.)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)

	Name	Description
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329].
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value.
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683]
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329].
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329].

Extension Methods

	Name	Description
	PollValuesAnnotated(IObservable.Unit.) [▶ 882]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence (Defined by ValueSymbolExtensions [▶ 877].)
	PollValuesAnnotated(TimeSpan) [▶ 883]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time. (Defined by ValueSymbolExtensions [▶ 877].)
	WhenValueChanged [▶ 884]	Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object.) [▶ 888]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception.) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., CancellationToken) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object.,	Overloaded.

	Name	Description
	Action.Exception., CancellationToken) [▶ 891]	Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.1 DynamicSymbol.Equals Method

Equals

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool Equals(
    Object obj
)
```

VB

```
Public Overrides Function Equals (
    obj As Object
) As Boolean
```

Parameters

obj	Type: System.Object The object to compare with the current object.

Return Value

Type: Boolean
true if the specified Object is equal to this instance; otherwise, false.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.2 DynamicSymbol.GetDynamicMemberNames Method

Returns the enumeration of all dynamic member names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override IEnumerable<string> GetDynamicMemberNames()
```

VB

```
Public Overrides Function GetDynamicMemberNames As IEnumerable(Of String)
```

Return Value

Type: IEnumerable.String.
A sequence that contains dynamic member names.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.3 DynamicSymbol.GetHashCode Method

Gets the HashCode of the Address

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override int GetHashCode()
```

VB

```
Public Overrides Function GetHashCode As Integer
```

Return Value

Type: Int32
A hash code for this instance, suitable for use in hashing algorithms and data structures like a hash table.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.4 DynamicSymbol.OnReadAnyValue Method

Handler function for reading ADS 'Any' Values.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual Object OnReadAnyValue(
    Type managedType
)
```

VB

```
Protected Overridable Function OnReadAnyValue (
    managedType As Type
) As Object
```

Parameters

managedType	Type: System.Type Managed type to read.

Return Value

Type: Object
System.Object.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.5 DynamicSymbol.OnReadRawValue Method

Handler function for reading Raw symbol value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual byte[] OnReadRawValue(
    int timeout
)
```

VB

```
Protected Overridable Function OnReadRawValue (
    timeout As Integer
) As Byte()
```

Parameters

timeout	Type: System.Int32 The timeout.

Return Value

Type: .Byte.
System.Byte[].

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.6 DynamicSymbol.OnReadValue Method

Handler function for the

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual Object OnReadValue(
    int timeout
)
```

VB

```
Protected Overridable Function OnReadValue (
    timeout As Integer
) As Object
```

Parameters

timeout	Type: System.Int32 The timeout.

Return Value

Type: Object
System.Object.

Exceptions

Exception	Condition
CannotAccessVirtualSymbolException [▶ 31]	

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.7 DynamicSymbol.OnSetInstanceName Method

Sets a new InstanceName InstancePath

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual void OnSetInstanceName(
    string instanceName
)
```

VB

```
Protected Overridable Sub OnSetInstanceName (
    instanceName As String
)
```

Parameters

instanceName	Type: System.String Instance name.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.8 DynamicSymbol.OnTryReadValue Method

Handler function for the

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual int OnTryReadValue(
    int timeout,
    out Object value
)
```

VB

```
Protected Overridable Function OnTryReadValue (
    timeout As Integer,
    <OutAttribute> ByRef value As Object
) As Integer
```

Parameters

timeout	Type: System.Int32 The timeout.
value	Type: System.Object. The value.

Return Value

Type: Int32
The error Code.

Exceptions

Exception	Condition
CannotAccessVirtualSymbolException [▶ 31]	

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.9 DynamicSymbol.OnTryWriteValue Method

Handler Function for writing value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual int OnTryWriteValue(
    Object value,
    int timeout
)
```

VB

```
Protected Overridable Function OnTryWriteValue (
    value As Object,
    timeout As Integer
) As Integer
```

Parameters

value	Type: System.Object The value.
timeout	Type: System.Int32 The timeout.

Return Value

Type: Int32

Exceptions

Exception	Condition
CannotAccessVirtualSymbolException [▶ 31]	

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.10 DynamicSymbol.OnWriteRawValue Method

Handler function for reading symbols raw value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual void OnWriteRawValue(
    byte[] rawValue,
    int timeout
)
```

VB

```
Protected Overridable Sub OnWriteRawValue (
    rawValue As Byte(),
    timeout As Integer
)
```

Parameters

rawValue	Type: .System.Byte. The raw value.
timeout	Type: System.Int32 The timeout.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.11 DynamicSymbol.OnWriteValue Method

Handler Function for writing value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual void OnWriteValue(
    Object value,
    int timeout
)
```

VB

```
Protected Overridable Sub OnWriteValue (
    value As Object,
    timeout As Integer
)
```

Parameters

value	Type: System.Object The value.
timeout	Type: System.Int32 The timeout.

Exceptions

Exception	Condition
CannotAccessVirtualSymbolException [▶ 31]	

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.12 DynamicSymbol.ReadAnyValue Method

Reads the value of this Value [▶ 1683] into a new created instance of the managed type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadAnyValue(
    Type managedType
)
```

VB

```
Public Function ReadAnyValue (
    managedType As Type
) As Object
```

Parameters

managedType	Type: System.Type The tp.

Return Value

Type: Object
Read value (System.Object).

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueAnySymbol.WriteAnyValue(Object) [▶ 1673]

IValueAnySymbol.UpdateAnyValue(Object.) [▶ 1672]

5.8.17.2.13 DynamicSymbol.ReadRawValue Method

Overload List

	Name	Description
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicSymbol.ReadRawValue Method

Reads the Symbols raw value

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public byte[] ReadRawValue()
```

VB

```
Public Function ReadRawValue As Byte()
```

Field Value

Type: .Byte.
The raw value in bytes.

Return Value

Type: .Byte.
System.Byte[].

Implements

IValueRawSymbol.ReadRawValue. [▶ 1679]

Reference

DynamicSymbol Class [▶ 1329]

ReadRawValue Overload [▶ 1364]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicSymbol.ReadRawValue Method (Int32)

Reads the Symbols raw value

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public byte[] ReadRawValue(
    int timeout
)
```

VB

```
Public Function ReadRawValue (
    timeout As Integer
) As Byte()
```

Parameters

timeout	Type: System.Int32 The timeout in ms.

Field Value

Type: .Byte.
The raw value in bytes.

Return Value

Type: .Byte.
System.Byte[].

Implements

IValueRawSymbol.ReadRawValue(Int32) [▶ 1680]

Remarks

A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

DynamicSymbol Class [▶ 1329]

ReadRawValue Overload [▶ 1364]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.14 DynamicSymbol.ReadValue Method

Overload List

	Name	Description
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329].

	Name	Description
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329].

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicSymbol.ReadValue Method

Reads the value of this DynamicSymbol [▶ 1329].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadValue()
```

VB

```
Public Function ReadValue As Object
```

Return Value

Type: Object
System.Object.

Implements

IValueSymbol.ReadValue. [▶ 1690]

Remarks

Examples

Dynamic Read access

```
namespace Sample
{
    using System;
    using System.Diagnostics;
    using System.Threading;
    using TwinCAT;
    using TwinCAT.Ads;
    using TwinCAT.Ads.TypeSystem;
    using TwinCAT.Ads.ValueAccess;
    using TwinCAT.TypeSystem;
    using TwinCAT.TypeSystem.Generic;

    class SymbolBrowserProgramV2DynamicTree
    {

    #region CODE_SAMPLE_SIMPLEDYNAMIC
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
```

```
        // Get the AdsAddress from command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        using (TcAdsClient client = new TcAdsClient())
        {
        // Connect to the target device
        client.Connect(address);

        // Usage of "dynamic" Type and Symbols (>= .NET4 only)
        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.DynamicTree);
        IAdsSymbolLoader dynLoader = (IAdsSymbolLoader)SymbolLoaderFactory.Create(client, settings);

        #endregion

        // Set the Default setting for Notifications
        dynLoader.DefaultNotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 200,
 2000);

        // Get the Symbols (Dynamic Symbols)
        dynamic dynamicSymbols = ((IDynamicSymbolLoader)dynLoader).SymbolsDynamic;

        dynamic adsPort = dynamicSymbols.TwinCAT_SystemInfoVarList._AppInfo.AdsPort;

        #region CODE_SAMPLE_SIMPLEDYNAMIC

        // Access Main Symbol with Dynamic Language Runtime support (DLR)
        // Dynamically created property "Main"
        //dynamic symMain = dynamicSymbols.Main;

        // Main is an 'VirtualSymbol' / Organizational unit that doesn't have a value
        // Calling ReadValue is not allowed
        //bool test = symMain.HasValue;
        //dynamic invalid = symMain.ReadValue();

        //Reading TaskInfo Value
        //With calling ReadValue() a 'snapshot' of the Symbols Instance is taken
        dynamic vTaskInfoArray = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo.ReadValue();

        // Getting the Snapshot time in UTC format
        DateTime timeStamp1 = vTaskInfoArray.UtcTimeStamp;

        // Getting TaskInfo Symbol for Task 1
        dynamic symTaskInfo1 = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo[1];

        // Getting CycleCount Symbol
        dynamic symCycleCount = symTaskInfo1.CycleCount;


        // Take Snapshot value of the ApplicationInfo struct
        dynamic vAppInfo = dynamicSymbols.TwinCAT_SystemInfoVarList._AppInfo.ReadValue();
        // Get the UTC Timestamp of the snapshot

        DateTime timeStamp2 = vAppInfo.UtcTimeStamp;

        // Access the ProjectName of the ApplicationInfo Snapshot (type-safe!)
        string projectNameValue = vAppInfo.ProjectName;

        // Reading the CycleCount Value
        uint cycleCountValue = symTaskInfo1.CycleCount.ReadValue();     // Taking a Value Snapshot
        #endregion

        // Registering for dynamic "ValueChanged" events for the Values
        // Using Default Notification settings
        symCycleCount.ValueChanged += new EventHandler<ValueChangedArgs>(cycleCount_ValueChanged);

        // Override default notification settings
        symTaskInfo1.NotificationSettings = new NotificationSettings(AdsTransMode.Cyclic,500,0);

        // Register for ValueChanged event.
        symTaskInfo1.ValueChanged += new EventHandler<ValueChangedArgs>(taskInfo1Value_ValueChanged)
; // Struct Type

        Thread.Sleep(10000); // Sleep main thread for 10 Seconds
        }
        Console.WriteLine("CycleCount Changed events received: {0}",_cycleCountEvents);
        Console.WriteLine("taskInfo1 Changed events received: {0}", _taskInfo1Events);

        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
```

```
    }

    static object _notificationSynchronizer = new object();
    static int _cycleCountEvents = 0;

    /// <summary>
    /// Handler function for the CycleCount ValueChanged event.
    /// </summary>
    /// <param name="sender">The sender.</param>
    /// <param name="e">The e.</param>
    static void cycleCount_ValueChanged(object sender, ValueChangedArgs e)
    {
        lock(_notificationSynchronizer)
        {
        Interlocked.Increment(ref _cycleCountEvents);
        // val is a type safe value of int!
        dynamic val = e.Value;
        uint intVal = val;

        DateTime changedTime = e.UtcRtime.ToLocalTime(); // Convert UTC to local time
        Console.WriteLine("CycleCount changed to: {0}, TimeStamp: {1}", intVal, changedTime.ToString
("HH:mm:ss:fff"));
        }
    }

    static int _taskInfo1Events = 0;

    /// <summary>
    /// Handler function for the TaskInfo ValueChanged event.
    /// </summary>
    /// <param name="sender">The sender.</param>
    /// <param name="e">The e.</param>
    static void taskInfo1Value_ValueChanged(object sender, ValueChangedArgs e)
    {
        lock (_notificationSynchronizer)
        {
        Interlocked.Increment(ref _taskInfo1Events);
        dynamic val = e.Value;
        DateTime changedTime = e.UtcRtime.ToLocalTime(); // Convert to local time

        // Val is a during Runtime created struct type and contains
        // the same Properties as related PLC object.
        int cycleTime = val.CycleTime;
        Console.WriteLine("TaskInfo1Value changed TimeStamp: {0}", changedTime.ToString("HH:mm:ss:ff
f"));
        }
    }
    }
}
```

Reference

DynamicSymbol Class [▶ 1329]

ReadValue Overload [▶ 1365]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicSymbol.ReadValue Method (Int32)

Reads the value of this DynamicSymbol [▶ 1329].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ReadValue(
    int timeout
)
```

VB

```
Public Function ReadValue (
    timeout As Integer
) As Object
```

Parameters

timeout	Type: System.Int32 The timeout in ms.

Field Value

Type: Object
The value.

Return Value

Type: Object
System.Object.

Implements

IValueSymbol.ReadValue(Int32) [▶ 1691]

Remarks

Calling on primitive types, a call of this method will return the primitive value. On complex types (structures and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: the raw byte Array will be returned, in dynamic mode: A Value will be created on the fly. A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

DynamicSymbol Class [▶ 1329]

ReadValue Overload [▶ 1365]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.15 DynamicSymbol.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.16 DynamicSymbol.TryGetMember Method

Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetMember(
    GetMemberBinder binder,
    out Object result
)
```

VB

```
Public Overrides Function TryGetMember (
    binder As GetMemberBinder,
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member on which the dynamic operation is performed. For example, for the Console.WriteLine(sampleObject.SampleProperty) statement, where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
result	Type: System.Object. The result of the get operation. For example, if the method is called for a property, you can assign the property value to result.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.17 DynamicSymbol.TryReadValue Method

Reads the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int TryReadValue(
    int timeout,
    out Object value
)
```

VB

```
Public Function TryReadValue (
    timeout As Integer,
    <OutAttribute> ByRef value As Object
) As Integer
```

Parameters

timeout	Type: System.Int32 The timeout in ms.
value	Type: System.Object. The symbol value.

Return Value

Type: Int32
The error code.

Implements

IValueSymbol3.TryReadValue(Int32, Object.) [▶ 1706]

Remarks

Calling on primitive types, a call of this method will return the primitive value. On complex types (structures and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: the raw byte Array will be returned, in dynamic mode: A Value will be created on the fly. A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.18 DynamicSymbol.TryWriteValue Method

Writes the specified value to the DynamicSymbol [▶ 1329].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int TryWriteValue(
    Object value,
    int timeout
)
```

VB

```
Public Function TryWriteValue (
    value As Object,
    timeout As Integer
) As Integer
```

Parameters

value	Type: System.Object The value.
timeout	Type: System.Int32 The timeout in ms.

Return Value

Type: Int32
The error code.

Implements

IValueSymbol3.TryWriteValue(Object, Int32) [▶ 1707]

Remarks

Calling on primitive types, a call of this method will directly write this Value. On complex types (structs and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: Only byte Arrays (of correct size) can be written) in dynamic mode: A Value that represents the value will be accepted also. A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Examples

Dynamic Write access

```
namespace Sample
{
    using System;
    using System.Diagnostics;
    using System.Threading;
    using TwinCAT;
    using TwinCAT.Ads;
    using TwinCAT.Ads.TypeSystem;
    using TwinCAT.Ads.ValueAccess;
    using TwinCAT.TypeSystem;
    using TwinCAT.TypeSystem.Generic;

    class SymbolBrowserProgramV2DynamicTree
    {

    #region CODE_SAMPLE_SIMPLEDYNAMIC
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        // Get the AdsAddress from command-line arguments
```

```
        AmsAddress address = ArgParser.Parse(args);

        using (TcAdsClient client = new TcAdsClient())
        {
        // Connect to the target device
        client.Connect(address);

        // Usage of "dynamic" Type and Symbols (>= .NET4 only)
        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.DynamicTree);
        IAdsSymbolLoader dynLoader = (IAdsSymbolLoader)SymbolLoaderFactory.Create(client, settings);

        #endregion

        // Set the Default setting for Notifications
        dynLoader.DefaultNotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 200,
 2000);

        // Get the Symbols (Dynamic Symbols)
        dynamic dynamicSymbols = ((IDynamicSymbolLoader)dynLoader).SymbolsDynamic;

        dynamic adsPort = dynamicSymbols.TwinCAT_SystemInfoVarList._AppInfo.AdsPort;

        #region CODE_SAMPLE_SIMPLEDYNAMIC

        // Access Main Symbol with Dynamic Language Runtime support (DLR)
        // Dynamically created property "Main"
        //dynamic symMain = dynamicSymbols.Main;

        // Main is an 'VirtualSymbol' / Organizational unit that doesn't have a value
        // Calling ReadValue is not allowed
        //bool test = symMain.HasValue;
        //dynamic invalid = symMain.ReadValue();

        //Reading TaskInfo Value
        //With calling ReadValue() a 'snapshot' of the Symbols Instance is taken
        dynamic vTaskInfoArray = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo.ReadValue();

        // Getting the Snapshot time in UTC format
        DateTime timeStamp1 = vTaskInfoArray.UtcTimeStamp;

        // Getting TaskInfo Symbol for Task 1
        dynamic symTaskInfo1 = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo[1];

        // Getting CycleCount Symbol
        dynamic symCycleCount = symTaskInfo1.CycleCount;


        // Take Snapshot value of the ApplicationInfo struct
        dynamic vAppInfo = dynamicSymbols.TwinCAT_SystemInfoVarList._AppInfo.ReadValue();
        // Get the UTC Timestamp of the snapshot

        DateTime timeStamp2 = vAppInfo.UtcTimeStamp;

        // Access the ProjectName of the ApplicationInfo Snapshot (type-safe!)
        string projectNameValue = vAppInfo.ProjectName;

        // Reading the CycleCount Value
        uint cycleCountValue = symTaskInfo1.CycleCount.ReadValue();     // Taking a Value Snapshot
        #endregion

        // Registering for dynamic "ValueChanged" events for the Values
        // Using Default Notification settings
        symCycleCount.ValueChanged += new EventHandler<ValueChangedArgs>(cycleCount_ValueChanged);

        // Override default notification settings
        symTaskInfo1.NotificationSettings = new NotificationSettings(AdsTransMode.Cyclic,500,0);

        // Register for ValueChanged event.
        symTaskInfo1.ValueChanged += new EventHandler<ValueChangedArgs>(taskInfo1Value_ValueChanged)
; // Struct Type

        Thread.Sleep(10000); // Sleep main thread for 10 Seconds
        }
        Console.WriteLine("CycleCount Changed events received: {0}",_cycleCountEvents);
        Console.WriteLine("taskInfo1 Changed events received: {0}", _taskInfo1Events);

        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
    }
```

```
    static object _notificationSynchronizer = new object();
    static int _cycleCountEvents = 0;

    /// <summary>
    /// Handler function for the CycleCount ValueChanged event.
    /// </summary>
    /// <param name="sender">The sender.</param>
    /// <param name="e">The e.</param>
    static void cycleCount_ValueChanged(object sender, ValueChangedArgs e)
    {
        lock(_notificationSynchronizer)
        {
        Interlocked.Increment(ref _cycleCountEvents);
        // val is a type safe value of int!
        dynamic val = e.Value;
        uint intVal = val;

        DateTime changedTime = e.UtcRtime.ToLocalTime(); // Convert UTC to local time
        Console.WriteLine("CycleCount changed to: {0}, TimeStamp: {1}", intVal, changedTime.ToString
("HH:mm:ss:fff"));
        }
    }

    static int _taskInfo1Events = 0;

    /// <summary>
    /// Handler function for the TaskInfo ValueChanged event.
    /// </summary>
    /// <param name="sender">The sender.</param>
    /// <param name="e">The e.</param>
    static void taskInfo1Value_ValueChanged(object sender, ValueChangedArgs e)
    {
        lock (_notificationSynchronizer)
        {
        Interlocked.Increment(ref _taskInfo1Events);
        dynamic val = e.Value;
        DateTime changedTime = e.UtcRtime.ToLocalTime(); // Convert to local time

        // Val is a during Runtime created struct type and contains
        // the same Properties as related PLC object.
        int cycleTime = val.CycleTime;
        Console.WriteLine("TaskInfo1Value changed TimeStamp: {0}", changedTime.ToString("HH:mm:ss:ff
f"));
        }
    }
    }
}
```

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.19 DynamicSymbol.UpdateAnyValue Method

Reads the value of this Value [▶ 1683] into the specified managed value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void UpdateAnyValue(
    ref Object valueObject
)
```

VB

```
Public Sub UpdateAnyValue (
    ByRef valueObject As Object
)
```

Parameters

valueObject	Type: System.Object. The managed object.

Return Value

Type:
Read value (System.Object).

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueAnySymbol.ReadAnyValue(Type) [▶ 1670]

IValueAnySymbol.WriteAnyValue(Object) [▶ 1673]

5.8.17.2.20 DynamicSymbol.WriteAnyValue Method

Writes the value represented by the managed value to this Value [▶ 1683]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteAnyValue(
    Object managedValue
)
```

VB

```
Public Sub WriteAnyValue (
    managedValue As Object
)
```

Parameters

managedValue	Type: System.Object The managed value.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicSymbol.ReadAnyValue(Type) [▶ 1363]

DynamicSymbol.UpdateAnyValue(Object.) [▶ 1374]

5.8.17.2.21 DynamicSymbol.WriteRawValue Method

Overload List

	Name	Description
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicSymbol.WriteRawValue Method (.Byte.)

Writes the Symbol raw Value

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteRawValue(
    byte[] rawValue
)
```

VB

```
Public Sub WriteRawValue (
    rawValue As Byte()
)
```

Parameters

rawValue	Type: .System.Byte. The raw value.

Implements

IValueRawSymbol.WriteRawValue(.Byte.) [▶ 1681]

Reference

DynamicSymbol Class [▶ 1329]

WriteRawValue Overload [▶ 1376]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicSymbol.WriteRawValue Method (.Byte., Int32)

Writes the Symbol raw Value

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteRawValue(
    byte[] rawValue,
    int timeout
)
```

VB

```
Public Sub WriteRawValue (
    rawValue As Byte(),
    timeout As Integer
)
```

Parameters

rawValue	Type: .System.Byte. The raw value.
timeout	Type: System.Int32 The timeout.

Implements

IValueRawSymbol.WriteRawValue(.Byte., Int32) [▶ 1682]

Reference

DynamicSymbol Class [▶ 1329]

WriteRawValue Overload [▶ 1376]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.2.22 DynamicSymbol.WriteValue Method

Overload List

	Name	Description
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329].
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329].

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicSymbol.WriteValue Method (Object)

Writes the specified value to the DynamicSymbol [▶ 1329].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteValue(
    Object value
)
```

VB

```
Public Sub WriteValue (
    value As Object
)
```

Parameters

value	Type: System.Object The value.

Implements

IValueSymbol.WriteValue(Object) [▶ 1692]

Remarks

Examples

Dynamic Write access

```
namespace Sample
{
    using System;
    using System.Diagnostics;
    using System.Threading;
    using TwinCAT;
    using TwinCAT.Ads;
    using TwinCAT.Ads.TypeSystem;
    using TwinCAT.Ads.ValueAccess;
    using TwinCAT.TypeSystem;
    using TwinCAT.TypeSystem.Generic;

    class SymbolBrowserProgramV2DynamicTree
    {

    #region CODE_SAMPLE_SIMPLEDYNAMIC
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        // Get the AdsAddress from command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        using (TcAdsClient client = new TcAdsClient())
        {
        // Connect to the target device
        client.Connect(address);

        // Usage of "dynamic" Type and Symbols (>= .NET4 only)
        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.DynamicTree);
        IAdsSymbolLoader dynLoader = (IAdsSymbolLoader)SymbolLoaderFactory.Create(client, settings);
```

```
        #endregion

        // Set the Default setting for Notifications
        dynLoader.DefaultNotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 200,
 2000);

        // Get the Symbols (Dynamic Symbols)
        dynamic dynamicSymbols = ((IDynamicSymbolLoader)dynLoader).SymbolsDynamic;

        dynamic adsPort = dynamicSymbols.TwinCAT_SystemInfoVarList._AppInfo.AdsPort;

        #region CODE_SAMPLE_SIMPLEDYNAMIC

        // Access Main Symbol with Dynamic Language Runtime support (DLR)
        // Dynamically created property "Main"
        //dynamic symMain = dynamicSymbols.Main;

        // Main is an 'VirtualSymbol' / Organizational unit that doesn't have a value
        // Calling ReadValue is not allowed
        //bool test = symMain.HasValue;
        //dynamic invalid = symMain.ReadValue();

        //Reading TaskInfo Value
        //With calling ReadValue() a 'snapshot' of the Symbols Instance is taken
        dynamic vTaskInfoArray = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo.ReadValue();

        // Getting the Snapshot time in UTC format
        DateTime timeStamp1 = vTaskInfoArray.UtcTimeStamp;

        // Getting TaskInfo Symbol for Task 1
        dynamic symTaskInfo1 = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo[1];

        // Getting CycleCount Symbol
        dynamic symCycleCount = symTaskInfo1.CycleCount;


        // Take Snapshot value of the ApplicationInfo struct
        dynamic vAppInfo = dynamicSymbols.TwinCAT_SystemInfoVarList._AppInfo.ReadValue();
        // Get the UTC Timestamp of the snapshot

        DateTime timeStamp2 = vAppInfo.UtcTimeStamp;

        // Access the ProjectName of the ApplicationInfo Snapshot (type-safe!)
        string projectNameValue = vAppInfo.ProjectName;

        // Reading the CycleCount Value
        uint cycleCountValue = symTaskInfo1.CycleCount.ReadValue();     // Taking a Value Snapshot
        #endregion

        // Registering for dynamic "ValueChanged" events for the Values
        // Using Default Notification settings
        symCycleCount.ValueChanged += new EventHandler<ValueChangedArgs>(cycleCount_ValueChanged);

        // Override default notification settings
        symTaskInfo1.NotificationSettings = new NotificationSettings(AdsTransMode.Cyclic,500,0);

        // Register for ValueChanged event.
        symTaskInfo1.ValueChanged += new EventHandler<ValueChangedArgs>(taskInfo1Value_ValueChanged)
; // Struct Type

        Thread.Sleep(10000); // Sleep main thread for 10 Seconds
        }
        Console.WriteLine("CycleCount Changed events received: {0}",_cycleCountEvents);
        Console.WriteLine("taskInfo1 Changed events received: {0}", _taskInfo1Events);

        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
    }

    static object _notificationSynchronizer = new object();
    static int _cycleCountEvents = 0;

    /// <summary>
    /// Handler function for the CycleCount ValueChanged event.
    /// </summary>
    /// <param name="sender">The sender.</param>
    /// <param name="e">The e.</param>
    static void cycleCount_ValueChanged(object sender, ValueChangedArgs e)
```

```
    {
        lock(_notificationSynchronizer)
        {
        Interlocked.Increment(ref _cycleCountEvents);
        // val is a type safe value of int!
        dynamic val = e.Value;
        uint intVal = val;

        DateTime changedTime = e.UtcRtime.ToLocalTime(); // Convert UTC to local time
        Console.WriteLine("CycleCount changed to: {0}, TimeStamp: {1}", intVal, changedTime.ToString
("HH:mm:ss:fff"));
        }
    }

    static int _taskInfo1Events = 0;

    /// <summary>
    /// Handler function for the TaskInfo ValueChanged event.
    /// </summary>
    /// <param name="sender">The sender.</param>
    /// <param name="e">The e.</param>
    static void taskInfo1Value_ValueChanged(object sender, ValueChangedArgs e)
    {
        lock (_notificationSynchronizer)
        {
        Interlocked.Increment(ref _taskInfo1Events);
        dynamic val = e.Value;
        DateTime changedTime = e.UtcRtime.ToLocalTime(); // Convert to local time

        // Val is a during Runtime created struct type and contains
        // the same Properties as related PLC object.
        int cycleTime = val.CycleTime;
        Console.WriteLine("TaskInfo1Value changed TimeStamp: {0}", changedTime.ToString("HH:mm:ss:ff
f"));
        }
    }
    }
}
```

Reference

DynamicSymbol Class [▶ 1329]

WriteValue Overload [▶ 1377]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicSymbol.WriteValue Method (Object, Int32)

Writes the specified value to the DynamicSymbol [▶ 1329].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void WriteValue(
    Object value,
    int timeout
)
```

VB

```
Public Sub WriteValue (
    value As Object,
    timeout As Integer
)
```

Parameters

value	Type: System.Object The value.
timeout	Type: System.Int32 The timeout in ms.

Implements

IValueSymbol.WriteValue(Object, Int32) [▶ 1693]

Remarks

Calling on primitive types, a call of this method will directly write this Value. On complex types (structs and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: Only byte Arrays (of correct size) can be written) in dynamic mode: A Value that represents the value will be accepted also. A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Examples

Dynamic Write access

```
namespace Sample
{
    using System;
    using System.Diagnostics;
    using System.Threading;
    using TwinCAT;
    using TwinCAT.Ads;
    using TwinCAT.Ads.TypeSystem;
    using TwinCAT.Ads.ValueAccess;
    using TwinCAT.TypeSystem;
    using TwinCAT.TypeSystem.Generic;

    class SymbolBrowserProgramV2DynamicTree
    {

    #region CODE_SAMPLE_SIMPLEDYNAMIC
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        // Get the AdsAddress from command-line arguments
        AmsAddress address = ArgParser.Parse(args);

        using (TcAdsClient client = new TcAdsClient())
        {
        // Connect to the target device
        client.Connect(address);

        // Usage of "dynamic" Type and Symbols (>= .NET4 only)
        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.DynamicTree);
        IAdsSymbolLoader dynLoader = (IAdsSymbolLoader)SymbolLoaderFactory.Create(client, settings);

        #endregion

        // Set the Default setting for Notifications
        dynLoader.DefaultNotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 200,
 2000);

        // Get the Symbols (Dynamic Symbols)
        dynamic dynamicSymbols = ((IDynamicSymbolLoader)dynLoader).SymbolsDynamic;

        dynamic adsPort = dynamicSymbols.TwinCAT_SystemInfoVarList._AppInfo.AdsPort;

        #region CODE_SAMPLE_SIMPLEDYNAMIC

        // Access Main Symbol with Dynamic Language Runtime support (DLR)
```

```
        // Dynamically created property "Main"
        //dynamic symMain = dynamicSymbols.Main;

        // Main is an 'VirtualSymbol' / Organizational unit that doesn't have a value
        // Calling ReadValue is not allowed
        //bool test = symMain.HasValue;
        //dynamic invalid = symMain.ReadValue();

        //Reading TaskInfo Value
        //With calling ReadValue() a 'snapshot' of the Symbols Instance is taken
        dynamic vTaskInfoArray = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo.ReadValue();

        // Getting the Snapshot time in UTC format
        DateTime timeStamp1 = vTaskInfoArray.UtcTimeStamp;

        // Getting TaskInfo Symbol for Task 1
        dynamic symTaskInfo1 = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo[1];

        // Getting CycleCount Symbol
        dynamic symCycleCount = symTaskInfo1.CycleCount;


        // Take Snapshot value of the ApplicationInfo struct
        dynamic vAppInfo = dynamicSymbols.TwinCAT_SystemInfoVarList._AppInfo.ReadValue();
        // Get the UTC Timestamp of the snapshot

        DateTime timeStamp2 = vAppInfo.UtcTimeStamp;

        // Access the ProjectName of the ApplicationInfo Snapshot (type-safe!)
        string projectNameValue = vAppInfo.ProjectName;

        // Reading the CycleCount Value
        uint cycleCountValue = symTaskInfo1.CycleCount.ReadValue();     // Taking a Value Snapshot
        #endregion

        // Registering for dynamic "ValueChanged" events for the Values
        // Using Default Notification settings
        symCycleCount.ValueChanged += new EventHandler<ValueChangedArgs>(cycleCount_ValueChanged);

        // Override default notification settings
        symTaskInfo1.NotificationSettings = new NotificationSettings(AdsTransMode.Cyclic,500,0);

        // Register for ValueChanged event.
        symTaskInfo1.ValueChanged += new EventHandler<ValueChangedArgs>(taskInfo1Value_ValueChanged)
; // Struct Type

        Thread.Sleep(10000); // Sleep main thread for 10 Seconds
        }
        Console.WriteLine("CycleCount Changed events received: {0}",_cycleCountEvents);
        Console.WriteLine("taskInfo1 Changed events received: {0}", _taskInfo1Events);

        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
    }

    static object _notificationSynchronizer = new object();
    static int _cycleCountEvents = 0;

    /// <summary>
    /// Handler function for the CycleCount ValueChanged event.
    /// </summary>
    /// <param name="sender">The sender.</param>
    /// <param name="e">The e.</param>
    static void cycleCount_ValueChanged(object sender, ValueChangedArgs e)
    {
        lock(_notificationSynchronizer)
        {
        Interlocked.Increment(ref _cycleCountEvents);
        // val is a type safe value of int!
        dynamic val = e.Value;
        uint intVal = val;

        DateTime changedTime = e.UtcRtime.ToLocalTime(); // Convert UTC to local time
        Console.WriteLine("CycleCount changed to: {0}, TimeStamp: {1}", intVal, changedTime.ToString
("HH:mm:ss:fff"));
        }
    }

    static int _taskInfo1Events = 0;
```

```
    /// <summary>
    /// Handler function for the TaskInfo ValueChanged event.
    /// </summary>
    /// <param name="sender">The sender.</param>
    /// <param name="e">The e.</param>
    static void taskInfo1Value_ValueChanged(object sender, ValueChangedArgs e)
    {
        lock (_notificationSynchronizer)
        {
        Interlocked.Increment(ref _taskInfo1Events);
        dynamic val = e.Value;
        DateTime changedTime = e.UtcRtime.ToLocalTime(); // Convert to local time

        // Val is a during Runtime created struct type and contains
        // the same Properties as related PLC object.
        int cycleTime = val.CycleTime;
        Console.WriteLine("TaskInfo1Value changed TimeStamp: {0}", changedTime.ToString("HH:mm:ss:ff
f"));
        }
    }
    }
}
```

Reference

DynamicSymbol Class [▶ 1329]

WriteValue Overload [▶ 1377]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.3 DynamicSymbol Events

The DynamicSymbol [▶ 1329] type exposes the following members.

Events

	Name	Description
⚡	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed.
⚡	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.3.1 DynamicSymbol.RawValueChanged Event

Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event EventHandler<RawValueChangedArgs> RawValueChanged
```

VB

```
Public Event RawValueChanged As EventHandler(Of RawValueChangedArgs)
```

Value

Type: System.EventHandler.RawValueChangedArgs [▶ 1725].

Implements

IValueRawSymbol.RawValueChanged [▶ 1683]

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.3.2 DynamicSymbol.ValueChanged Event

Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public event EventHandler<ValueChangedArgs> ValueChanged
```

VB

```
Public Event ValueChanged As EventHandler(Of ValueChangedArgs)
```

Value

Type: System.EventHandler.ValueChangedArgs [▶ 1847].

Implements

IValueSymbol.ValueChanged [▶ 1694]

Remarks

Examples

Use Dynamic Notifications

```
using System;
using System.Diagnostics;
using System.Threading;
using TwinCAT;
using TwinCAT.Ads;
using TwinCAT.Ads.TypeSystem;
using TwinCAT.Ads.ValueAccess;
using TwinCAT.TypeSystem;
using TwinCAT.TypeSystem.Generic;

namespace Sample
{
    class SymbolBrowserV2Notifications
```

```
    {
    /// <summary>
    /// Defines the entry point of the application.
    /// </summary>
    /// <param name="args">The arguments.</param>
    static void Main(string[] args)
    {
        // Parse the Command Line Parameters.
        AmsAddress address = ArgParser.Parse(args);

        #region DEFAULTNOTIFICATON_SAMPLE

        // Create AdsClient object
        using (TcAdsClient client = new TcAdsClient())
        {
        // No automatic Synchronization (necessary for Console applications without message loop)
        client.Synchronize = false;

        // Connect to client
        client.Connect(address);

        // Usage of 'dynamic' type/symbol loader
        SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.DynamicTree, ValueA
ccessMode.IndexGroupOffsetPreferred);
        IAdsSymbolLoader dynLoader = (IAdsSymbolLoader)SymbolLoaderFactory.Create(client, settings);

        // Set the DefaultNotification Properties
        dynLoader.DefaultNotificationSettings = new NotificationSettings(AdsTransMode.ClientOnChange
, 200, 2000);

        // Determine the symbols
        dynamic dynamicSymbols = ((IDynamicSymbolLoader)dynLoader).SymbolsDynamic;

        // Task 1 Symbol (build in symbol)
        dynamic task1Symbol = dynamicSymbols.TwinCAT_SystemInfoVarList._TaskInfo[1];

        // CycleCount Symbol
        dynamic cycleCountSymbol = task1Symbol.CycleCount;

        // Override Notification Setting for Cycle Count Symbol
        cycleCountSymbol.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 250,
 0);

        // Register Dynamic Value Changed event.
        cycleCountSymbol.ValueChanged += new EventHandler<ValueChangedArgs>(cycleCount_ValueChanged)
;
#endregion
        // Sleep main thread to receive notifications
        Thread.Sleep(10000);
#region DEFAULTNOTIFICATON_SAMPLE
        }
#endregion
        Console.WriteLine("CycleCount Changed events received: {0}", _cycleCountEvents);
        Console.WriteLine("");
        Console.WriteLine("Press [Enter] for leave:");
        Console.ReadLine();
    }

    /// <summary>
    /// The cycle count event counter
    /// </summary>
    static int _cycleCountEvents = 0;

    /// <summary>
    /// Handler function for CycleCount changed events.
    /// </summary>
    /// <param name="sender">Event sender.</param>
    /// <param name="args">Event arguments.</param>
    static void cycleCount_ValueChanged(object sender, ValueChangedArgs args)
    {
        Interlocked.Increment(ref _cycleCountEvents);

        // Use Value as dynamic (type safe: UINT)  object.
        dynamic val = args.Value;
        uint intVal = val;

        DateTime changedTime = args.UtcRtime.ToLocalTime(); // Convert UTC to local time
        Console.WriteLine("CycleCount changed to: {0}, TimeStamp: {1}", intVal, changedTime.ToString
("HH:mm:ss:fff"));
```

```
    }
    }
}
```

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.4 DynamicSymbol Operators

The DynamicSymbol [▶ 1329] type exposes the following members.

Operators

	Name	Description
	Equality [▶ 1386]	Operator==
	Inequality [▶ 1387]	Implements the != operator.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.4.1 DynamicSymbol.Equality Operator

Operator==

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator ==(
    DynamicSymbol o1,
    DynamicSymbol o2
)
```

VB

```
Public Shared Operator = (
    o1 As DynamicSymbol,
    o2 As DynamicSymbol
) As Boolean
```

Parameters

o1	Type: TwinCAT.TypeSystem.DynamicSymbol [▶ 1329] The o1.
o2	Type: TwinCAT.TypeSystem.DynamicSymbol [▶ 1329] The o2.

Return Value

Type: Boolean
The result of the operator.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.4.2 DynamicSymbol.Inequality Operator

Implements the != operator.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator !=(
    DynamicSymbol o1,
    DynamicSymbol o2
)
```

VB

```
Public Shared Operator <> (
    o1 As DynamicSymbol,
    o2 As DynamicSymbol
) As Boolean
```

Parameters

o1	Type: TwinCAT.TypeSystem.DynamicSymbol [▶ 1329] The o1.
o2	Type: TwinCAT.TypeSystem.DynamicSymbol [▶ 1329] The o2.

Return Value

Type: Boolean
The result of the operator.

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.5 DynamicSymbol Fields

The DynamicSymbol [▶ 1329] type exposes the following members.

Fields

	Name	Description
	normalizedName [▶ 1388]	The normalized name of this .

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.17.5.1 DynamicSymbol.normalizedName Field

The normalized name of this .

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected string normalizedName
```

VB

```
Protected normalizedName As String
```

Field Value

Type: String

Reference

DynamicSymbol Class [▶ 1329]

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]

5.8.18 DynamicSymbolsContainer Class

Dynamic (Expandable) Symbols collection.

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicSymbolsContainer
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class DynamicSymbolsContainer : DynamicObject,
    IEnumerable<ISymbol>, IEnumerable
```

VB

```
Public NotInheritable Class DynamicSymbolsContainer
    Inherits DynamicObject
    Implements IEnumerable(Of ISymbol), IEnumerable
```

The DynamicSymbolsContainer type exposes the following members.

Constructors

	Name	Description
	DynamicSymbolsContainer [▶ 1390]	Initializes a new instance of the DynamicSymbolsContainer class (for internal use only)

Properties

	Name	Description
	Item [▶ 1391]	Gets the DynamicSymbol [▶ 1329] with the specified name.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1393]	Returns the enumeration of all dynamic member names. (Overrides DynamicObject.GetDynamicMemberNames..)
	GetEnumerator [▶ 1394]	Gets the enumerator.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)

	Name	Description
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetInstance [▶ 1394]	Tries to get the IInstance [▶ 1556]. of the specified path.
	TryGetMember [▶ 1395]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicObject.TryGetMember(GetMemberBinder, Object.).)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicObject.)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)

Remarks

The DynamicSymbolsContainer collection adds dynamically its child Symbols as Members (for access like "Main.Symbol")

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.18.1 DynamicSymbolsContainer Constructor

Initializes a new instance of the DynamicSymbolsContainer [▶ 1388] class (for internal use only)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DynamicSymbolsContainer(
    SymbolCollection<ISymbol> symbols
)
```

VB

```
Public Sub New (
    symbols As SymbolCollection(Of ISymbol)
)
```

Parameters

symbols	Type: TwinCAT.TypeSystem.Generic.SymbolCollection [▶ 1964].ISymbol [▶ 1634]. The symbols.

Exceptions

Exception	Condition
ArgumentNullException	symbols

Reference

DynamicSymbolsContainer Class [▶ 1388]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.18.2 DynamicSymbolsContainer Properties

The DynamicSymbolsContainer [▶ 1388] type exposes the following members.

Properties

	Name	Description
	Item [▶ 1391]	Gets the DynamicSymbol [▶ 1329] with the specified name.

Reference

DynamicSymbolsContainer Class [▶ 1388]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.18.2.1 DynamicSymbolsContainer.Item Property

Gets the DynamicSymbol [▶ 1329] with the specified name.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DynamicSymbol this[
    string name
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    name As String
) As DynamicSymbol
    Get
```

Parameters

name	Type: System.String The name.

Return Value

Type: DynamicSymbol [▶ 1329]
DynamicSymbol.

Exceptions

Exception	Condition
KeyNotFoundException	Symbol name not found in DynamicSymbols collection!

Reference

DynamicSymbolsContainer Class [▶ 1388]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.18.3 DynamicSymbolsContainer Methods

The DynamicSymbolsContainer [▶ 1388] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1393]	Returns the enumeration of all dynamic member names. (Overrides DynamicObject.GetDynamicMemberNames.)
	GetEnumerator [▶ 1394]	Gets the enumerator.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)

	Name	Description
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetInstance [▶ 1394]	Tries to get the IInstance [▶ 1556]. of the specified path.
	TryGetMember [▶ 1395]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicObject.TryGetMember(GetMemberBinder, Object.).)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicObject.)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)

Reference

DynamicSymbolsContainer Class [▶ 1388]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.18.3.1 DynamicSymbolsContainer.GetDynamicMemberNames Method

Returns the enumeration of all dynamic member names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override IEnumerable<string> GetDynamicMemberNames()
```

VB

```
Public Overrides Function GetDynamicMemberNames As IEnumerable(Of String)
```

Return Value

Type: IEnumerable.String.
A sequence that contains dynamic member names.

Reference

DynamicSymbolsContainer Class [▶ 1388]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.18.3.2 DynamicSymbolsContainer.GetEnumerator Method

Gets the enumerator.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumerator<ISymbol> GetEnumerator()
```

VB

```
Public Function GetEnumerator As IEnumerator(Of ISymbol)
```

Return Value

Type: IEnumerator.ISymbol [▶ 1634].
A IEnumerator.T. that can be used to iterate through the collection.

Implements

IEnumerable.T..GetEnumerator.

Reference

DynamicSymbolsContainer Class [▶ 1388]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.18.3.3 DynamicSymbolsContainer.TryGetInstance Method

Tries to get the IInstance [▶ 1556]. of the specified path.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetInstance(
    string instanceSpecifier,
    out ISymbol symbol
)
```

VB

```
Public Function TryGetInstance (
    instanceSpecifier As String,
    <OutAttribute> ByRef symbol As ISymbol
) As Boolean
```

Parameters

instanceSpecifier	Type: System.String The instance path or Instance Name (dependent of Mode [▶ 1894] setting)
symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634]. The symbol.

Return Value

Type: Boolean
true if the IInstance [▶ 1556] is found; otherwise, false

Exceptions

Exception	Condition
ArgumentNullException	instancePath
ArgumentException	

Reference

DynamicSymbolsContainer Class [▶ 1388]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.18.3.4 DynamicSymbolsContainer.TryGetMember Method

Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetMember(
    GetMemberBinder binder,
    out Object result
)
```

VB

```
Public Overrides Function TryGetMember (
    binder As GetMemberBinder,
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member on which the dynamic operation is performed. For example, for the Console.WriteLine(sampleObject.SampleProperty) statement, where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
result	Type: System.Object. The result of the get operation. For example, if the method is called for a property, you can assign the property value to result.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicSymbolsContainer Class [▶ 1388]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.19 DynamicUnionInstance Class

Dynamic union instance

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]
      TwinCAT.TypeSystem.DynamicUnionInstance

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class DynamicUnionInstance : DynamicSymbol,
    IUnionInstance, ISymbol, IAttributedInstance, IInstance, IBitSize
```

VB

```
Public NotInheritable Class DynamicUnionInstance
    Inherits DynamicSymbol
    Implements IUnionInstance, ISymbol, IAttributedInstance, IInstance,
    IBitSize
```

The DynamicUnionInstance type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	FieldInstances [▶ 1402]	Gets the member instances of the Struct Instance [▶ 1615].
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1404]	Returns the enumeration of all dynamic member names. (Overrides DynamicSymbol.GetDynamicMemberNames. [▶ 1356].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)

	Name	Description
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetMember [▶ 1405]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicSymbol.TryGetMember(GetMemberBinder, Object.) [▶ 1370].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember [▶ 1406]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Overrides DynamicObject.TrySetMember(SetMemberBinder, Object).)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.19.1 DynamicUnionInstance Properties

The DynamicUnionInstance [▶ 1396] type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	FieldInstances [▶ 1402]	Gets the member instances of the Struct Instance [▶ 1615].
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicUnionInstance Class [▶ 1396]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.19.1.1 DynamicUnionInstance.FieldInstances Property

Gets the member instances of the Struct Instance [▶ 1615].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlySymbolCollection FieldInstances { get; }
```

VB

```
Public ReadOnly Property FieldInstances As ReadOnlySymbolCollection
    Get
```

Property Value

Type: ReadOnlySymbolCollection [▶ 1772]
The member instances.

Implements

IUnionInstance.FieldInstances [▶ 1656]

Reference

DynamicUnionInstance Class [▶ 1396]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.19.2 DynamicUnionInstance Methods

The DynamicUnionInstance [▶ 1396] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1404]	Returns the enumeration of all dynamic member names. (Overrides DynamicSymbol.GetDynamicMemberNames. [▶ 1356].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetMember [▶ 1405]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicSymbol.TryGetMember(GetMemberBinder, Object.) [▶ 1370].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)

	Name	Description
	TrySetMember [▶ 1406]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Overrides DynamicObject.TrySetMember(SetMemberBinder, Object).)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicUnionInstance Class [▶ 1396]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.19.2.1 DynamicUnionInstance.GetDynamicMemberNames Method

Returns the enumeration of all dynamic member names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override IEnumerable<string> GetDynamicMemberNames()
```

VB

```
Public Overrides Function GetDynamicMemberNames As IEnumerable(Of String)
```

Return Value

Type: IEnumerable.String.
A sequence that contains dynamic member names.

Reference

DynamicUnionInstance Class [▶ 1396]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.19.2.2 DynamicUnionInstance.TryGetMember Method

Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetMember(
    GetMemberBinder binder,
    out Object result
)
```

VB

```
Public Overrides Function TryGetMember (
    binder As GetMemberBinder,
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member on which the dynamic operation is performed. For example, for the Console.WriteLine(sampleObject.SampleProperty) statement, where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
result	Type: System.Object. The result of the get operation. For example, if the method is called for a property, you can assign the property value to result.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicUnionInstance Class [▶ 1396]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.19.2.3 DynamicUnionInstance.TrySetMember Method

Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TrySetMember(
    SetMemberBinder binder,
    Object value
)
```

VB

```
Public Overrides Function TrySetMember (
    binder As SetMemberBinder,
    value As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.SetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member to which the value is being assigned. For example, for the statement sampleObject.SampleProperty = "Test", where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
value	Type: System.Object The value to set to the member. For example, for sampleObject.SampleProperty = "Test", where sampleObject is an instance of the class derived from the DynamicObject class, the value is "Test".

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.)

Reference

DynamicUnionInstance Class [▶ 1396]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.19.3 DynamicUnionInstance Events

The DynamicUnionInstance [▶ 1396] type exposes the following members.

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicUnionInstance Class [▶ 1396]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20 DynamicValue Class

Dynamic value (uses RuntimeBinding for ISymbol [▶ 1634] value reading / writing).

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicValue
      TwinCAT.TypeSystem.DynamicPointerValue [▶ 1274]
      TwinCAT.TypeSystem.DynamicReferenceValue [▶ 1293]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class DynamicValue : DynamicObject,
    IValue, IStructValue, IArrayValue
```

VB

```
Public Class DynamicValue
    Inherits DynamicObject
    Implements IValue, IStructValue, IArrayValue
```

The DynamicValue type exposes the following members.

Properties

	Name	Description
	Age [▶ 1411]	Gets the age of the value (last successful read of the value)
	CachedRaw [▶ 1412]	Gets the cached Raw internal Data.
	DataType [▶ 1412]	Gets the data type bound to this IValue [▶ 1659]
	IsPrimitive [▶ 1413]	Gets a value indicating whether this IValue [▶ 1659] is a primitive value.
	ResolvedType [▶ 1413]	Gets the resolved type.
	Symbol [▶ 1414]	Gets the symbol that is bound to this value.

	Name	Description
	UtcTimeStamp [▶ 1414]	Gets the Time stamp of the last successful read of the Value.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1417]	Returns the enumeration of all dynamic member names. (Overrides DynamicObject.GetDynamicMemberNames..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Read [▶ 1418]	Reads the value (via ADS)
	ReadMember [▶ 1418]	Reads the specified member element.
	ResolveValue [▶ 1419]	Resolves the Value object to its primitive value.
	ToString [▶ 1419]	Returns a String that represents this instance. (Overrides Object.ToString..)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert [▶ 1420]	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Overrides DynamicObject.TryConvert(ConvertBinder, Object.).)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetArrayElementValues [▶ 1421]	Returns Array Element values.

	Name	Description
	TryGetIndex [▶ 1421]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Overrides DynamicObject.TryGetIndex(GetIndexBinder, .Object., Object.).)
	TryGetIndexValue(.Int32., Object.) [▶ 1423]	Reads the specified array element.
	TryGetIndexValue(.Object., Object.) [▶ 1423]	Tries the get index value.
	TryGetMember [▶ 1424]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicObject.TryGetMember(GetMemberBinder, Object.).)
	TryGetMemberValue [▶ 1425]	Tries the get member value.
	TryInvoke [▶ 1426]	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Overrides DynamicObject.TryInvoke(InvokeBinder, .Object., Object.).)
	TryInvokeMember [▶ 1427]	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Overrides DynamicObject.TryInvokeMember(InvokeMemberBinder, .Object., Object.).)
	TryResolveValue [▶ 1428]	Tries to resolves the Value object to its primitive value.
	TrySetIndex [▶ 1428]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Overrides DynamicObject.TrySetIndex(SetIndexBinder, .Object., Object).)
	TrySetIndexValue [▶ 1429]	Tries to set the indexed value on Arrays
	TrySetMember [▶ 1430]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Overrides DynamicObject.TrySetMember(SetMemberBinder, Object).)
	TrySetMemberValue [▶ 1431]	Tries to Set a Member/Property Value
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	Write [▶ 1432]	Writes the value (via ADS)
	WriteMember [▶ 1432]	Writes the specified member element.

Fields

	Name	Description
	_symbol [▶ 1433]	Symbol that is bound to this value.
	valueFactory [▶ 1434]	The value factory

Remarks

The DynamicValue adds dynamic run time behaviour to the IValue [▶ 1659]Value/IValue [▶ 1659]. That means e.g. for struct values that .NET Properties are on-the-fly defined and dispatched at runtime just like defined in the structs structs data type definition. Another example is the access of Array Element values through indexes.

Examples

Sample for the dynamic resolution of Symbols and reading values:

Dynamic Symbol access

```
/// <summary>
/// Defines the entry point of the application.
/// </summary>
/// <param name="args">The arguments.</param>
static void Main(string[] args)
{
    // Get the AdsAddress from command-line arguments
    AmsAddress address = ArgParser.Parse(args);

    using (TcAdsClient client = new TcAdsClient())
    {
    // Connect to the target device
    client.Connect(address);

    // Usage of "dynamic" Type and Symbols (>= .NET4 only)
    SymbolLoaderSettings settings = new SymbolLoaderSettings(SymbolsLoadMode.DynamicTree);
    IAdsSymbolLoader dynLoader = (IAdsSymbolLoader)SymbolLoaderFactory.Create(client, settings);
```

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]

System.Dynamic.DynamicObject

TwinCAT.TypeSystem.IValue [▶ 1659]

5.8.20.1 DynamicValue Properties

The DynamicValue [▶ 1407] type exposes the following members.

Properties

	Name	Description
	Age [▶ 1411]	Gets the age of the value (last successful read of the value)
	CachedRaw [▶ 1412]	Gets the cached Raw internal Data.
	DataType [▶ 1412]	Gets the data type bound to this IValue [▶ 1659]
	IsPrimitive [▶ 1413]	Gets a value indicating whether this IValue [▶ 1659] is a primitive value.

	Name	Description
	ResolvedType [▶ 1413]	Gets the resolved type.
	Symbol [▶ 1414]	Gets the symbol that is bound to this value.
	UtcTimeStamp [▶ 1414]	Gets the Time stamp of the last successful read of the Value.

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.1.1 DynamicValue.Age Property

Gets the age of the value (last successful read of the value)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TimeSpan Age { get; }
```

VB

```
Public ReadOnly Property Age As TimeSpan
    Get
```

Property Value

Type: TimeSpan
The age.

Implements

IValue.Age [▶ 1660]

Exceptions

Exception	Condition
NotImplementedException	

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicValue.UtcTimeStamp [▶ 1414]

5.8.20.1.2 DynamicValue.CachedRaw Property

Gets the cached Raw internal Data.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public byte[] CachedRaw { get; }
```

VB

```
Public ReadOnly Property CachedRaw As Byte()
    Get
```

Property Value

Type: .Byte.
The raw cached data.

Implements

IValue.CachedRaw [▶ 1661]

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.1.3 DynamicValue.DataType Property

Gets the data type bound to this IValue [▶ 1659]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IDataType DataType { get; }
```

VB

```
Public ReadOnly Property DataType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the data.

Implements

IValue.DataType [▶ 1661]

Exceptions

Exception	Condition
NotImplementedException	

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.1.4 DynamicValue.IsPrimitive Property

Gets a value indicating whether this IValue [▶ 1659] is a primitive value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsPrimitive { get; }
```

VB

```
Public ReadOnly Property IsPrimitive As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is primitive; otherwise, false.

Implements

IValue.IsPrimitive [▶ 1662]

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.1.5 DynamicValue.ResolvedType Property

Gets the resolved type.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected IDataType ResolvedType { get; }
```

VB

```
Protected ReadOnly Property ResolvedType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
Resolved type.

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.1.6 DynamicValue.Symbol Property

Gets the symbol that is bound to this value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ISymbol Symbol { get; }
```

VB

```
Public ReadOnly Property Symbol As ISymbol
    Get
```

Property Value

Type: ISymbol [▶ 1634]
The symbol.

Implements

IValue.Symbol [▶ 1662]

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.1.7 DynamicValue.UtcTimeStamp Property

Gets the Time stamp of the last successful read of the Value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DateTime UtcTimeStamp { get; }
```

VB

```
Public ReadOnly Property UtcTimeStamp As DateTime
    Get
```

Property Value

Type: DateTime
The read time stamp.

Implements

IValue.UtcTimeStamp [▶ 1663]

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2 DynamicValue Methods

The DynamicValue [▶ 1407] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDynamicMemberNames [▶ 1417]	Returns the enumeration of all dynamic member names. (Overrides DynamicObject.GetDynamicMemberNames.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Read [▶ 1418]	Reads the value (via ADS)
	ReadMember [▶ 1418]	Reads the specified member element.

	Name	Description
	ResolveValue [▶ 1419]	Resolves the Value object to its primitive value.
	ToString [▶ 1419]	Returns a String that represents this instance. (Overrides Object.ToString..)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert [▶ 1420]	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Overrides DynamicObject.TryConvert(ConvertBinder, Object.).)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetArrayElementValues [▶ 1421]	Returns Array Element values.
	TryGetIndex [▶ 1421]	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Overrides DynamicObject.TryGetIndex(GetIndexBinder, .Object., Object.).)
	TryGetIndexValue(.Int32., Object.) [▶ 1423]	Reads the specified array element.
	TryGetIndexValue(.Object., Object.) [▶ 1423]	Tries the get index value.
	TryGetMember [▶ 1424]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Overrides DynamicObject.TryGetMember(GetMemberBinder, Object.).)
	TryGetMemberValue [▶ 1425]	Tries the get member value.
	TryInvoke [▶ 1426]	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Overrides DynamicObject.TryInvoke(InvokeBinder, .Object., Object.).)
	TryInvokeMember [▶ 1427]	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Overrides DynamicObject.TryInvokeMember(InvokeMemberBinder, .Object., Object.).)

	Name	Description
	TryResolveValue [▶ 1428]	Tries to resolves the Value object to its primitive value.
	TrySetIndex [▶ 1428]	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Overrides DynamicObject.TrySetIndex(SetIndexBinder, .Object., Object).)
	TrySetIndexValue [▶ 1429]	Tries to set the indexed value on Arrays
	TrySetMember [▶ 1430]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Overrides DynamicObject.TrySetMember(SetMemberBinder, Object).)
	TrySetMemberValue [▶ 1431]	Tries to Set a Member/Property Value
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	Write [▶ 1432]	Writes the value (via ADS)
	WriteMember [▶ 1432]	Writes the specified member element.

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.1 DynamicValue.GetDynamicMemberNames Method

Returns the enumeration of all dynamic member names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override IEnumerable<string> GetDynamicMemberNames()
```

VB

```
Public Overrides Function GetDynamicMemberNames As IEnumerable(Of String)
```

Return Value

Type: IEnumerable.String.
A sequence that contains dynamic member names.

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.2 DynamicValue.Read Method

Reads the value (via ADS)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Read()
```

VB

```
Public Sub Read
```

Implements

IValue.Read. [▶ 1664]

Exceptions

Exception	Condition
SymbolException [▶ 606]	

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.3 DynamicValue.ReadMember Method

Reads the specified member element.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual Object ReadMember(
    ISymbol memberInstance
)
```

VB

```
Protected Overridable Function ReadMember (
    memberInstance As ISymbol
) As Object
```

Parameters

memberInstance	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The member instance.

Return Value

Type: Object

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.4 DynamicValue.ResolveValue Method

Resolves the Value object to its primitive value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object ResolveValue(
    bool resolveEnumToPrimitive
)
```

VB

```
Public Function ResolveValue (
    resolveEnumToPrimitive As Boolean
) As Object
```

Parameters

resolveEnumToPrimitive	Type: System.Boolean if set to trueIEnumValue [▶ 1550]s are resolved to their primitives also.

Return Value

Type: Object
System.Object.

Implements

IValue.ResolveValue(Boolean) [▶ 1664]

Remarks

If the value is not primitive, this method returns the IValue [▶ 1659] itself.

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.5 DynamicValue.ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.6 DynamicValue.TryConvert Method

Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryConvert(
    ConvertBinder binder,
    out Object result
)
```

VB

```
Public Overrides Function TryConvert (
    binder As ConvertBinder,
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.ConvertBinder Provides information about the conversion operation. The binder.Type property provides the type to which the object must be converted. For example, for the statement (String)sampleObject in C# (CType(sampleObject, Type) in Visual Basic), where sampleObject is an instance of the class derived from the DynamicObject class, binder.Type returns the String type. The binder.Explicit property provides information about the kind of conversion that occurs. It returns true for explicit conversion and false for implicit conversion.
result	Type: System.Object. The result of the type conversion operation.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.)

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.7 DynamicValue.TryGetArrayElementValues Method

Returns Array Element values.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetArrayElementValues(
    out IEnumerable<Object> elementValues
)
```

VB

```
Public Function TryGetArrayElementValues (
    <OutAttribute> ByRef elementValues As IEnumerable(Of Object)
) As Boolean
```

Parameters

elementValues	Type: System.Collections.Generic.IEnumerable.Object.. The element values.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Implements

IArrayValue.TryGetArrayElementValues(IEnumerable.Object..) [▶ 1510]

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.8 DynamicValue.TryGetIndex Method

Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetIndex(
    GetIndexBinder binder,
    Object[] indexes,
    out Object result
)
```

VB

```
Public Overrides Function TryGetIndex (
    binder As GetIndexBinder,
    indexes As Object(),
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetIndexBinder Provides information about the operation.
indexes	Type: .System.Object. The indexes that are used in the operation. For example, for the sampleObject[3] operation in C# (sampleObject(3) in Visual Basic), where sampleObject is derived from the DynamicObject class, indexes[0] is equal to 3.
result	Type: System.Object. The result of the index operation.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.9 DynamicValue.TryGetIndexValue Method

Overload List

	Name	Description
	TryGetIndexValue(.Int32., Object.) [▶ 1423]	Reads the specified array element.
	TryGetIndexValue(.Object., Object.) [▶ 1423]	Tries the get index value.

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicValue.TryGetIndexValue Method (.Int32., Object.)

Reads the specified array element.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetIndexValue(
    int[] indices,
    out Object value
)
```

VB

```
Public Function TryGetIndexValue (
    indices As Integer(),
    <OutAttribute> ByRef value As Object
) As Boolean
```

Parameters

indices	Type: .System.Int32. The indices.
value	Type: System.Object. The value.

Return Value

Type: Boolean
System.Object.

Implements

IArrayValue.TryGetIndexValue(.Int32., Object.) [▶ 1510]

Reference

DynamicValue Class [▶ 1407]

TryGetIndexValue Overload [▶ 1422]

TwinCAT.TypeSystem Namespace [▶ 1184]

DynamicValue.TryGetIndexValue Method (.Object., Object.)

Tries the get index value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetIndexValue(
    Object[] indexes,
    out Object result
)
```

VB

```
Public Function TryGetIndexValue (
    indexes As Object(),
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

indexes	Type: .System.Object. The indexes.
result	Type: System.Object. The result.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

DynamicValue Class [▶ 1407]

TryGetIndexValue Overload [▶ 1422]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.10 DynamicValue.TryGetMember Method

Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryGetMember(
    GetMemberBinder binder,
    out Object result
)
```

VB

```
Public Overrides Function TryGetMember (
    binder As GetMemberBinder,
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.GetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member on which the dynamic operation is performed. For example, for the Console.WriteLine(sampleObject.SampleProperty) statement, where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
result	Type: System.Object. The result of the get operation. For example, if the method is called for a property, you can assign the property value to result.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a run-time exception is thrown.)

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.11 DynamicValue.TryGetMemberValue Method

Tries the get member value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual bool TryGetMemberValue(
    string name,
    out Object result
)
```

VB

```
Public Overridable Function TryGetMemberValue (
    name As String,
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

name	Type: System.String The name.
result	Type: System.Object. The result.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Implements

IStructValue.TryGetMemberValue(String, Object.) [▶ 1626]

Exceptions

Exception	Condition
SymbolException [▶ 606]	

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.12 DynamicValue.TryInvoke Method

Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryInvoke(
    InvokeBinder binder,
    Object[] args,
    out Object result
)
```

VB

```
Public Overrides Function TryInvoke (
    binder As InvokeBinder,
    args As Object(),
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.InvokeBinder Provides information about the invoke operation.
args	Type: .System.Object. The arguments that are passed to the object during the invoke operation. For example, for the sampleObject(100) operation, where sampleObject is derived from the DynamicObject class, args is equal to 100.
result	Type: System.Object. The result of the object invocation.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.13 DynamicValue.TryInvokeMember Method

Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TryInvokeMember(
    InvokeMemberBinder binder,
    Object[] args,
    out Object result
)
```

VB

```
Public Overrides Function TryInvokeMember (
    binder As InvokeMemberBinder,
    args As Object(),
    <OutAttribute> ByRef result As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.InvokeMemberBinder Provides information about the dynamic operation. The binder.Name property provides the name of the member on which the dynamic operation is performed. For example, for the statement sampleObject.SampleMethod(100), where sampleObject is an instance of the class derived from the DynamicObject class, binder.Name returns "SampleMethod". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
args	Type: .System.Object. The arguments that are passed to the object member during the invoke operation. For example, for the statement sampleObject.SampleMethod(100), where sampleObject is derived from the DynamicObject class, args[0] is equal to 100.
result	Type: System.Object. The result of the member invocation.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.)

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.14 DynamicValue.TryResolveValue Method

Tries to resolves the Value object to its primitive value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryResolveValue(
    bool resolveEnumToPrimitive,
    out Object value
)
```

VB

```
Public Function TryResolveValue (
    resolveEnumToPrimitive As Boolean,
    <OutAttribute> ByRef value As Object
) As Boolean
```

Parameters

resolveEnumToPrimitive	Type: System.Boolean if set to trueIEnumValue [▶ 1550]s are resolved to their primitives also.
value	Type: System.Object. The value.

Return Value

Type: Boolean
true if value can be resolved, false otherwise.

Implements

IValue.TryResolveValue(Boolean, Object.) [▶ 1665]

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.15 DynamicValue.TrySetIndex Method

Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TrySetIndex(
    SetIndexBinder binder,
    Object[] indexes,
    Object value
)
```

VB

```
Public Overrides Function TrySetIndex (
    binder As SetIndexBinder,
    indexes As Object(),
    value As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.SetIndexBinder Provides information about the operation.
indexes	Type: .System.Object. The indexes that are used in the operation. For example, for the sampleObject[3] = 10 operation in C# (sampleObject(3) = 10 in Visual Basic), where sampleObject is derived from the DynamicObject class, indexes[0] is equal to 3.
value	Type: System.Object The value to set to the object that has the specified index. For example, for the sampleObject[3] = 10 operation in C# (sampleObject(3) = 10 in Visual Basic), where sampleObject is derived from the DynamicObject class, value is equal to 10.

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.16 DynamicValue.TrySetIndexValue Method

Tries to set the indexed value on Arrays

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TrySetIndexValue(
    Object[] indexes,
    Object value
)
```

VB

```
Public Function TrySetIndexValue (
    indexes As Object(),
    value As Object
) As Boolean
```

Parameters

indexes	Type: .System.Object. The indexes.
value	Type: System.Object The value.

Return Value

Type: Boolean
true if succeeded, false otherwise.

Implements

IArrayValue.TrySetIndexValue(.Object., Object) [▶ 1511]

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.17 DynamicValue.TrySetMember Method

Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool TrySetMember(
    SetMemberBinder binder,
    Object value
)
```

VB

```
Public Overrides Function TrySetMember (
    binder As SetMemberBinder,
    value As Object
) As Boolean
```

Parameters

binder	Type: System.Dynamic.SetMemberBinder Provides information about the object that called the dynamic operation. The binder.Name property provides the name of the member to which the value is being assigned. For example, for the statement sampleObject.SampleProperty = "Test", where sampleObject is an instance of the class derived from the

	DynamicObject class, binder.Name returns "SampleProperty". The binder.IgnoreCase property specifies whether the member name is case-sensitive.
value	Type: System.Object The value to set to the member. For example, for sampleObject.SampleProperty = "Test", where sampleObject is an instance of the class derived from the DynamicObject class, the value is "Test".

Return Value

Type: Boolean
true if the operation is successful; otherwise, false. If this method returns false, the run-time binder of the language determines the behavior. (In most cases, a language-specific run-time exception is thrown.)

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.18 DynamicValue.TrySetMemberValue Method

Tries to Set a Member/Property Value

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TrySetMemberValue(
    string name,
    Object value
)
```

VB

```
Public Function TrySetMemberValue (
    name As String,
    value As Object
) As Boolean
```

Parameters

name	Type: System.String The name of the member
value	Type: System.Object The value.

Return Value

Type: Boolean
true if succeeded, otherwise false otherwise.

Implements

IStructValue.TrySetMemberValue(String, Object) [▶ 1627]

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.19 DynamicValue.Write Method

Writes the value (via ADS)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Write()
```

VB

```
Public Sub Write
```

Implements

IValue.Write. [▶ 1665]

Exceptions

Exception	Condition
SymbolException [▶ 606]	

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.2.20 DynamicValue.WriteMember Method

Writes the specified member element.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected virtual void WriteMember(
    ISymbol memberInstance,
    Object value
)
```

VB

```
Protected Overridable Sub WriteMember (
    memberInstance As ISymbol,
    value As Object
)
```

Parameters

memberInstance	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The member instance.
value	Type: System.Object The value.

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.3 DynamicValue Fields

The DynamicValue [▶ 1407] type exposes the following members.

Fields

	Name	Description
	_symbol [▶ 1433]	Symbol that is bound to this value.
	valueFactory [▶ 1434]	The value factory

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.3.1 DynamicValue._symbol Field

Symbol that is bound to this value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected ISymbol _symbol
```

VB

```
Protected _symbol As ISymbol
```

Field Value

Type: ISymbol [▶ 1634]

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.20.3.2 DynamicValue.valueFactory Field

The value factory

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected IAccessorValueFactory valueFactory
```

VB

```
Protected valueFactory As IAccessorValueFactory
```

Field Value

Type: IAccessorValueFactory

Reference

DynamicValue Class [▶ 1407]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.21 DynamicVirtualStructInstance Class

Dynamic struct instance

Inheritance Hierarchy

System.Object
  System.Dynamic.DynamicObject
    TwinCAT.TypeSystem.DynamicSymbol [▶ 1329]
      TwinCAT.TypeSystem.DynamicStructInstance [▶ 1315]
        TwinCAT.TypeSystem.DynamicVirtualStructInstance
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public sealed class DynamicVirtualStructInstance : DynamicStructInstance,
    IVirtualStructInstance, IStructInstance, ISymbol, IAttributedInstance, IInstance,
    IBitSize
```

VB

```
Public NotInheritable Class DynamicVirtualStructInstance
    Inherits DynamicStructInstance
    Implements IVirtualStructInstance, IStructInstance, ISymbol, IAttributedInstance,
    IInstance, IBitSize
```

The DynamicVirtualStructInstance type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	HasRpcMethods [▶ 1322]	Gets a value indicating whether this instance has RPC methods (Inherited from DynamicStructInstance [▶ 1315].)
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	MemberInstances [▶ 1322]	Gets the member instances of the Struct Instance [▶ 1615]. (Inherited from DynamicStructInstance [▶ 1315].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Methods

	Name	Description
	AddMember [▶ 1442]	Adds an member instance.
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1326]	Returns the enumeration of all dynamic member names. (Inherited from DynamicStructInstance [▶ 1315].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetMember [▶ 1326]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Inherited from DynamicStructInstance [▶ 1315].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember [▶ 1327]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicStructInstance [▶ 1315].)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.21.1 DynamicVirtualStructInstance Properties

The DynamicVirtualStructInstance [▶ 1434] type exposes the following members.

Properties

	Name	Description
	_InnerSymbol [▶ 1336]	Inner symbol object wrapped by this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	AccessRights [▶ 1337]	Gets the access rights. (Inherited from DynamicSymbol [▶ 1329].)
	Attributes [▶ 1337]	Gets the Symbol Attributes (Inherited from DynamicSymbol [▶ 1329].)
	BitSize [▶ 1338]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from DynamicSymbol [▶ 1329].)
	ByteSize [▶ 1338]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from DynamicSymbol [▶ 1329].)
	Category [▶ 1339]	Gets the category. (Inherited from DynamicSymbol [▶ 1329].)
	Comment [▶ 1339]	Gets the comment of the IInstance [▶ 1556] (Inherited from DynamicSymbol [▶ 1329].)
	Connection [▶ 1340]	Gets the connection bound to this DynamicSymbol [▶ 1329] (Inherited from DynamicSymbol [▶ 1329].)
	ContextMask [▶ 1341]	Gets the context mask. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	DataType [▶ 1341]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)
	HasRpcMethods [▶ 1322]	Gets a value indicating whether this instance has RPC methods (Inherited from DynamicStructInstance [▶ 1315].)
	HasValue [▶ 1342]	Gets a value indicating whether this instance has value (only the non VirtualInstances, what means the Symbols with locations). (Inherited from DynamicSymbol [▶ 1329].)
	InstanceName [▶ 1342]	Gets the name of the instance (without periods (.) (Inherited from DynamicSymbol [▶ 1329].)
	InstancePath [▶ 1343]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from DynamicSymbol [▶ 1329].)
	IsBitType [▶ 1343]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from DynamicSymbol [▶ 1329].)
	IsByteAligned [▶ 1344]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from DynamicSymbol [▶ 1329].)
	IsContainerType [▶ 1344]	Gets a value indicating whether this Symbol is acontainer type. (Inherited from DynamicSymbol [▶ 1329].)
	IsPersistent [▶ 1345]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from DynamicSymbol [▶ 1329].)
	IsPointer [▶ 1346]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsPrimitiveType [▶ 1346]	Gets a value indicating whether this instance is a primitive type. (Inherited from DynamicSymbol [▶ 1329].)
	IsReadOnly [▶ 1347]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from DynamicSymbol [▶ 1329].)
	IsRecursive [▶ 1347]	Gets a value indicating whether this instance is recursive. (Inherited from DynamicSymbol [▶ 1329].)
	IsReference [▶ 1348]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from DynamicSymbol [▶ 1329].)
	IsStatic [▶ 1348]	Gets a value indicating whether this instance is static. (Inherited from DynamicSymbol [▶ 1329].)
	MemberInstances [▶ 1322]	Gets the member instances of the Struct Instance [▶ 1615]. (Inherited from DynamicStructInstance [▶ 1315].)
	NormalizedName [▶ 1349]	Gets the normalized instance name (fixed name for dynamic property access that doesn't contain invalid characters), (Inherited from DynamicSymbol [▶ 1329].)
	NotificationSettings [▶ 1350]	Gets the notification settings. (Inherited from DynamicSymbol [▶ 1329].)
	Parent [▶ 1350]	Gets the parent Symbol (Inherited from DynamicSymbol [▶ 1329].)
	Size [▶ 1351]	Gets the size of the IInstance [▶ 1556] in bytes. (Inherited from DynamicSymbol [▶ 1329].)
	SubSymbols [▶ 1351]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from DynamicSymbol [▶ 1329].)
	TypeName [▶ 1352]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicVirtualStructInstance Class [▶ 1434]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.21.2 DynamicVirtualStructInstance Methods

The DynamicVirtualStructInstance [▶ 1434] type exposes the following members.

Methods

	Name	Description
	AddMember [▶ 1442]	Adds an member instance.
	Equals [▶ 1356]	Equals (Inherited from DynamicSymbol [▶ 1329].)
	GetDynamicMemberNames [▶ 1326]	Returns the enumeration of all dynamic member names. (Inherited from DynamicStructInstance [▶ 1315].)
	GetHashCode [▶ 1357]	Gets the HashCode of the Address (Inherited from DynamicSymbol [▶ 1329].)
	GetMetaObject	Provides a DynamicMetaObject that dispatches to the dynamic virtual methods. The object can be encapsulated inside another DynamicMetaObject to provide custom behavior for individual actions. This method supports the Dynamic Language Runtime infrastructure for language implementers and it is not intended to be used directly from your code. (Inherited from DynamicObject.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ReadAnyValue [▶ 1363]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue. [▶ 1364]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadRawValue(Int32) [▶ 1365]	Reads the Symbols raw value (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue. [▶ 1366]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ReadValue(Int32) [▶ 1368]	Reads the value of this DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	ToString [▶ 1369]	Returns a String that represents this instance. (Inherited from DynamicSymbol [▶ 1329].)
	TryBinaryOperation	Provides implementation for binary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as addition and multiplication. (Inherited from DynamicObject.)
	TryConvert	Provides implementation for type conversion operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that convert an object from one type to another. (Inherited from DynamicObject.)
	TryCreateInstance	Provides the implementation for operations that initialize a new instance of a dynamic object. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)

	Name	Description
	TryDeleteIndex	Provides the implementation for operations that delete an object by index. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryDeleteMember	Provides the implementation for operations that delete an object member. This method is not intended for use in C# or Visual Basic. (Inherited from DynamicObject.)
	TryGetIndex	Provides the implementation for operations that get a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for indexing operations. (Inherited from DynamicObject.)
	TryGetMember [▶ 1326]	Provides the implementation for operations that get member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as getting a value for a property. (Inherited from DynamicStructInstance [▶ 1315].)
	TryInvoke	Provides the implementation for operations that invoke an object. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as invoking an object or a delegate. (Inherited from DynamicObject.)
	TryInvokeMember	Provides the implementation for operations that invoke a member. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as calling a method. (Inherited from DynamicObject.)
	TryReadValue [▶ 1371]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	TrySetIndex	Provides the implementation for operations that set a value by index. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations that access objects by a specified index. (Inherited from DynamicObject.)
	TrySetMember [▶ 1327]	Provides the implementation for operations that set member values. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as setting a value for a property. (Inherited from DynamicStructInstance [▶ 1315].)
	TryUnaryOperation	Provides implementation for unary operations. Classes derived from the DynamicObject class can override this method to specify dynamic behavior for operations such as negation, increment, or decrement. (Inherited from DynamicObject.)
	TryWriteValue [▶ 1371]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)
	UpdateAnyValue [▶ 1374]	Reads the value of this Value [▶ 1683] into the specified managed value. (Inherited from DynamicSymbol [▶ 1329].)
	WriteAnyValue [▶ 1375]	Writes the value represented by the managed value to this Value [▶ 1683] (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte.) [▶ 1376]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteRawValue(.Byte., Int32) [▶ 1377]	Writes the Symbol raw Value (Inherited from DynamicSymbol [▶ 1329].)
	WriteValue(Object) [▶ 1378]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

	Name	Description
	WriteValue(Object, Int32) [▶ 1380]	Writes the specified value to the DynamicSymbol [▶ 1329]. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicVirtualStructInstance Class [▶ 1434]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.21.2.1 DynamicVirtualStructInstance.AddMember Method

Adds an member instance.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool AddMember(
    ISymbol memberInstance,
    IVirtualStructInstance parent
)
```

VB

```
Public Function AddMember (
    memberInstance As ISymbol,
    parent As IVirtualStructInstance
) As Boolean
```

Parameters

memberInstance	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The member instance.
parent	Type: TwinCAT.TypeSystem.IVirtualStructInstance [▶ 1708] The parent struct instance. Usually the this pointer.

Return Value

Type: Boolean

Implements

IVirtualStructInstance.AddMember(ISymbol, IVirtualStructInstance) [▶ 1711]

Reference

DynamicVirtualStructInstance Class [▶ 1434]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.21.3 DynamicVirtualStructInstance Events

The DynamicVirtualStructInstance [▶ 1434] type exposes the following members.

Events

	Name	Description
	RawValueChanged [▶ 1383]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)
	ValueChanged [▶ 1384]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from DynamicSymbol [▶ 1329].)

Reference

DynamicVirtualStructInstance Class [▶ 1434]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.22 EnumValue.T. Class

Enum Value

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.EnumValue.T.
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class EnumValue<T> : IEnumValue
where T : IConvertible
```

VB

```
Public Class EnumValue(Of T As IConvertible)
    Implements IEnumValue
```

Type Parameters

T	Enum base type (byte,sbyte,short,ushort,int,uint,long or ulong)

The EnumValue.T. type exposes the following members.

Properties

	Name	Description
	ManagedBaseType [▶ 1444]	Gets the enumeration base type (sint,byte,short,ushort,int,uint,Int64,UInt64 supported)
	Name [▶ 1445]	Gets the name of the Enum Value
	Primitive [▶ 1446]	Gets the value.
	RawValue [▶ 1446]	Gets the raw value of the enumeration (as byte array)
	Size [▶ 1447]	Gets the size of the Enum value (in bytes)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
S	Parse [▶ 1448]	Parse Enum Type string
	ToString [▶ 1449]	Returns a String that represents this instance. (Overrides Object.ToString..)
S	TryParse [▶ 1449]	Parse EnumType string

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.22.1 EnumValue.T. Properties

The EnumValue.T. [▶ 1443] generic type exposes the following members.

Properties

	Name	Description
	ManagedBaseType [▶ 1444]	Gets the enumeration base type (sint,byte,short,ushort,int,uint,Int64,UInt64 supported)
	Name [▶ 1445]	Gets the name of the Enum Value
	Primitive [▶ 1446]	Gets the value.
	RawValue [▶ 1446]	Gets the raw value of the enumeration (as byte array)
	Size [▶ 1447]	Gets the size of the Enum value (in bytes)

Reference

EnumValue.T. Class [▶ 1443]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.22.1.1 EnumValue.T..ManagedBaseType Property

Gets the enumeration base type (sint,byte,short,ushort,int,uint,Int64,UInt64 supported)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Type ManagedBaseType { get; }
```

VB

```
Public ReadOnly Property ManagedBaseType As Type
    Get
```

Property Value

Type: Type
The type of the base.

Implements

IEnumValue.ManagedBaseType [▶ 1551]

Reference

EnumValue.T. Class [▶ 1443]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.22.1.2 EnumValue.T..Name Property

Gets the name of the Enum Value

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Name { get; }
```

VB

```
Public ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Implements

IEnumValue.Name [▶ 1552]

Reference

EnumValue.T. Class [▶ 1443]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.22.1.3 EnumValue.T..Primitive Property

Gets the value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T Primitive { get; }
```

VB

```
Public ReadOnly Property Primitive As T
    Get
```

Property Value

Type: T [▶ 1443]
The value.

Reference

EnumValue.T. Class [▶ 1443]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.22.1.4 EnumValue.T..RawValue Property

Gets the raw value of the enumeration (as byte array)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public byte[] RawValue { get; }
```

VB

```
Public ReadOnly Property RawValue As Byte()
    Get
```

Property Value

Type: .Byte.
The raw value.

Implements

IEnumValue.RawValue [▶ 1553]

Exceptions

Exception	Condition
NotSupportedException	Base type of enum is not allowed!

Reference

EnumValue.T. Class [▶ 1443]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.22.1.5 EnumValue.T..Size Property

Gets the size of the Enum value (in bytes)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Size { get; }
```

VB

```
Public ReadOnly Property Size As Integer
    Get
```

Property Value

Type: Int32
The size.

Implements

IEnumValue.Size [▶ 1553]

Reference

EnumValue.T. Class [▶ 1443]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.22.2 EnumValue.T. Methods

The EnumValue.T. [▶ 1443] generic type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Parse [▶ 1448]	Parse Enum Type string

	Name	Description
	ToString [▶ 1449]	Returns a String that represents this instance. (Overrides Object.ToString..)
	TryParse [▶ 1449]	Parse EnumType string

Reference

EnumValue.T. Class [▶ 1443]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.22.2.1 EnumValue.T..Parse Method

Parse Enum Type string

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static EnumValue<T> Parse(
    IEnumType<T> type,
    string str
)
```

VB

```
Public Shared Function Parse (
    type As IEnumType(Of T),
    str As String
) As EnumValue(Of T)
```

Parameters

type	Type: TwinCAT.TypeSystem.IEnumType [▶ 1543].T [▶ 1443]. The type.
str	Type: System.String The string.

Return Value

Type: EnumValue [▶ 1443].T [▶ 1443].
EnumValue<T>.

Exceptions

Exception	Condition
FormatException	

Reference

EnumValue.T. Class [▶ 1443]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.22.2.2 EnumValue.T..ToString Method

Returns a String that represents this instance.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override string ToString()
```

VB

```
Public Overrides Function ToString As String
```

Return Value

Type: String
A String that represents this instance.

Reference

EnumValue.T. Class [▶ 1443]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.22.2.3 EnumValue.T..TryParse Method

Parse EnumType string

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool TryParse(
    IEnumType<T> type,
    string str,
    out EnumValue<T> value
)
```

VB

```
Public Shared Function TryParse (
    type As IEnumType(Of T),
    str As String,
    <OutAttribute> ByRef value As EnumValue(Of T)
) As Boolean
```

Parameters

type	Type: TwinCAT.TypeSystem.IEnumType [▶ 1543].T [▶ 1443]. The type.
str	Type: System.String The string.
value	Type: TwinCAT.TypeSystem.EnumValue [▶ 1443].T [▶ 1443].. The value.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

EnumValue.T. Class [▶ 1443]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23 EnumValueCollection Class

Class EnumValueCollection.

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.EnumValueCollection
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class EnumValueCollection : IList<IEnumValue>,
    ICollection<IEnumValue>, IEnumerable<IEnumValue>, IEnumerable
```

VB

```
Public Class EnumValueCollection
    Implements IList(Of IEnumValue), ICollection(Of IEnumValue),
    IEnumerable(Of IEnumValue), IEnumerable
```

The EnumValueCollection type exposes the following members.

Properties

	Name	Description
	Count [▶ 1452]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1452]	Gets a value indicating whether the ICollection.T. is read-only.
	Item.Int32. [▶ 1453]	Gets or sets the element at the specified index.
	Item.String. [▶ 1454]	Gets or sets the element at the specified index.

Methods

	Name	Description
	Add [▶ 1456]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1457]	Gets as read only.
	Clear [▶ 1457]	Removes all items from the ICollection.T..
	Contains(Object) [▶ 1458]	Determines whether [contains] [the specified value].

	Name	Description
	Contains(String) [▶ 1459]	Determines whether [contains] [the specified name].
	Contains(IEnumValue) [▶ 1459]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1460]	Copies the entire list.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1461]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetNames [▶ 1461]	Gets the names.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	GetValues [▶ 1462]	Gets the values.
	IndexOf [▶ 1462]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1463]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Parse [▶ 1463]	Parses the specified string to the Enum value.
	Remove [▶ 1464]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1465]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInfo [▶ 1466]	Tries the get information.
	TryParse [▶ 1466]	Parse the specified string to the enum value.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.1 EnumValueCollection Properties

The EnumValueCollection [▶ 1450] type exposes the following members.

Properties

	Name	Description
	Count [▶ 1452]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1452]	Gets a value indicating whether the ICollection.T. is read-only.
	Item.Int32. [▶ 1453]	Gets or sets the element at the specified index.

	Name	Description
	Item.String. [▶ 1454]	Gets or sets the element at the specified index.

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.1.1 EnumValueCollection.Count Property

Gets the number of elements contained in the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Count { get; }
```

VB

```
Public ReadOnly Property Count As Integer
    Get
```

Property Value

Type: Int32
The count.

Implements

ICollection.T..Count

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.1.2 EnumValueCollection.IsReadOnly Property

Gets a value indicating whether the ICollection.T. is read-only.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is read only; otherwise, false.

Implements

ICollection.T..IsReadOnly

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.1.3 EnumValueCollection.Item Property

Overload List

	Name	Description
	Item.Int32. [▶ 1453]	Gets or sets the element at the specified index.
	Item.String. [▶ 1454]	Gets or sets the element at the specified index.

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

EnumValueCollection.Item Property (Int32)

Gets or sets the element at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumValue this[
    int index
] { get; set; }
```

VB

```
Public Default Property Item (
    index As Integer
) As IEnumValue
    Get
    Set
```

Parameters

index — Type: System.Int32
The index.

Return Value

Type: IEnumValue [▶ 1550]
EnumValue<T>.

Implements

IList.T..Item.Int32.

Exceptions

Exception	Condition
NotImplementedException	

Reference

EnumValueCollection Class [▶ 1450]

Item Overload [▶ 1453]

TwinCAT.TypeSystem Namespace [▶ 1184]

EnumValueCollection.Item Property (String)

Gets or sets the element at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumValue this[
    string name
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    name As String
) As IEnumValue
    Get
```

Parameters

name	Type: System.String The name of the value

Return Value

Type: IEnumValue [▶ 1550]
EnumValue<T>.

Exceptions

Exception	Condition
NotImplementedException	
NotImplementedException	

Reference

EnumValueCollection Class [▶ 1450]

Item Overload [▶ 1453]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2 EnumValueCollection Methods

The EnumValueCollection [▶ 1450] type exposes the following members.

Methods

	Name	Description
	Add [▶ 1456]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1457]	Gets as read only.
	Clear [▶ 1457]	Removes all items from the ICollection.T..
	Contains(Object) [▶ 1458]	Determines whether [contains] [the specified value].
	Contains(String) [▶ 1459]	Determines whether [contains] [the specified name].
	Contains(IEnumValue) [▶ 1459]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1460]	Copies the entire list.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1461]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetNames [▶ 1461]	Gets the names.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	GetValues [▶ 1462]	Gets the values.
	IndexOf [▶ 1462]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1463]	Inserts an item to the IList.T. at the specified index.

	Name	Description
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Parse [▶ 1463]	Parses the specified string to the Enum value.
	Remove [▶ 1464]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1465]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInfo [▶ 1466]	Tries the get information.
	TryParse [▶ 1466]	Parse the specified string to the enum value.

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.1 EnumValueCollection.Add Method

Adds an item to the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Add(
    IEnumValue item
)
```

VB

```
Public Sub Add (
    item As IEnumValue
)
```

Parameters

item	Type: TwinCAT.TypeSystem.IEnumValue [▶ 1550] The object to add to the ICollection.T..

Implements

ICollection.T..Add(T)

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.2 EnumValueCollection.AsReadOnly Method

Gets as read only.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyEnumValueCollection AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlyEnumValueCollection
```

Field Value

Type: ReadOnlyEnumValueCollection [▶ 1736]
As read only.

Return Value

Type: ReadOnlyEnumValueCollection [▶ 1736]
ReadOnlyEnumValueCollection.

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.3 EnumValueCollection.Clear Method

Removes all items from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Clear()
```

VB

```
Public Sub Clear
```

Implements

ICollection.T..Clear.

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.4 EnumValueCollection.Contains Method

Overload List

	Name	Description
	Contains(Object) [▶ 1458]	Determines whether [contains] [the specified value].
	Contains(String) [▶ 1459]	Determines whether [contains] [the specified name].
	Contains(IEnumValue) [▶ 1459]	Determines whether the ICollection.T. contains a specific value.

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

EnumValueCollection.Contains Method (Object)

Determines whether [contains] [the specified value].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    Object value
)
```

VB

```
Public Function Contains (
    value As Object
) As Boolean
```

Parameters

value	Type: System.Object The value.

Return Value

Type: Boolean
true if [contains] [the specified value]; otherwise, false.

Reference

EnumValueCollection Class [▶ 1450]

Contains Overload [▶ 1458]

TwinCAT.TypeSystem Namespace [▶ 1184]

EnumValueCollection.Contains Method (String)

Determines whether [contains] [the specified name].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    string name
)
```

VB

```
Public Function Contains (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name.

Return Value

Type: Boolean
true if [contains] [the specified name]; otherwise, false.

Reference

EnumValueCollection Class [▶ 1450]

Contains Overload [▶ 1458]

TwinCAT.TypeSystem Namespace [▶ 1184]

EnumValueCollection.Contains Method (IEnumValue)

Determines whether the ICollection.T. contains a specific value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    IEnumValue item
)
```

VB

```
Public Function Contains (
    item As IEnumValue
) As Boolean
```

Parameters

item	Type: TwinCAT.TypeSystem.IEnumValue [▶ 1550] The object to locate in the ICollection.T..

Return Value

Type: Boolean
true if item is found in the ICollection.T.; otherwise, false.

Implements

ICollection.T..Contains(T)

Reference

EnumValueCollection Class [▶ 1450]

Contains Overload [▶ 1458]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.5 EnumValueCollection.CopyTo Method

Copies the entire list.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void CopyTo(
    IEnumValue[] array,
    int arrayIndex
)
```

VB

```
Public Sub CopyTo (
    array As IEnumValue(),
    arrayIndex As Integer
)
```

Parameters

array	Type: .TwinCAT.TypeSystem.IEnumValue [▶ 1550]. The array.
arrayIndex	Type: System.Int32 Index of the array.

Implements

ICollection.T..CopyTo(.T., Int32)

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.6 EnumValueCollection.GetEnumerator Method

Returns an enumerator that iterates through the collection.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumerator<IEnumValue> GetEnumerator()
```

VB

```
Public Function GetEnumerator As IEnumerator(Of IEnumValue)
```

Return Value

Type: IEnumerator.IEnumValue [▶ 1550].
A IEnumerator.T. that can be used to iterate through the collection.

Implements

IEnumerable.T..GetEnumerator.

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.7 EnumValueCollection.GetNames Method

Gets the names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string[] GetNames()
```

VB

```
Public Function GetNames As String()
```

Return Value

Type: .String.
System.String[].

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.8 EnumValueCollection.GetValues Method

Gets the values.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object[] GetValues()
```

VB

```
Public Function GetValues As Object()
```

Return Value

Type: .Object.
T[].

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.9 EnumValueCollection.IndexOf Method

Determines the index of a specific item in the IList.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int IndexOf(
    IEnumValue item
)
```

VB

```
Public Function IndexOf (
    item As IEnumValue
) As Integer
```

Parameters

item	Type: TwinCAT.TypeSystem.IEnumValue [▶ 1550] The object to locate in the IList.T..

Return Value

Type: Int32
The index of item if found in the list; otherwise, -1.

Implements

IList.T..IndexOf(T)

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.10 EnumValueCollection.Insert Method

Inserts an item to the IList.T. at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Insert(
    int index,
    IEnumValue item
)
```

VB

```
Public Sub Insert (
    index As Integer,
    item As IEnumValue
)
```

Parameters

index	Type: System.Int32 The zero-based index at which item should be inserted.
item	Type: TwinCAT.TypeSystem.IEnumValue [▶ 1550] The object to insert into the IList.T..

Implements

IList.T..Insert(Int32, T)

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.11 EnumValueCollection.Parse Method

Parses the specified string to the Enum value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object Parse(
    string name
)
```

VB

```
Public Function Parse (
    name As String
) As Object
```

Parameters

name	Type: System.String The name.

Return Value

Type: Object
T.

Exceptions

Exception	Condition
ArgumentOutOfRangeException	name

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.12 EnumValueCollection.Remove Method

Removes the first occurrence of a specific object from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Remove(
    IEnumValue item
)
```

VB

```
Public Function Remove (
    item As IEnumValue
) As Boolean
```

Parameters

item	Type: TwinCAT.TypeSystem.IEnumValue [▶ 1550] The object to remove from the ICollection.T..

Return Value

Type: Boolean
true if item was successfully removed from the ICollection.T.; otherwise, false. This method also returns false if item is not found in the original ICollection.T..

Implements

ICollection.T..Remove(T)

Exceptions

Exception	Condition
NotImplementedException	

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.13 EnumValueCollection.RemoveAt Method

Removes the IList.T. item at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void RemoveAt(
    int index
)
```

VB

```
Public Sub RemoveAt (
    index As Integer
)
```

Parameters

index	Type: System.Int32 The zero-based index of the item to remove.

Implements

IList.T..RemoveAt(Int32)

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.14 EnumValueCollection.TryGetInfo Method

Tries the get information.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetInfo(
    Object val,
    out IEnumValue ei
)
```

VB

```
Public Function TryGetInfo (
    val As Object,
    <OutAttribute> ByRef ei As IEnumValue
) As Boolean
```

Parameters

val	Type: System.Object The value.
ei	Type: TwinCAT.TypeSystem.IEnumValue [▶ 1550]. The ei.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.23.2.15 EnumValueCollection.TryParse Method

Parse the specified string to the enum value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryParse(
    string name,
    out Object value
)
```

VB

```
Public Function TryParse (
    name As String,
    <OutAttribute> ByRef value As Object
) As Boolean
```

Parameters

name	Type: System.String The name.
value	Type: System.Object. The value.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

EnumValueCollection Class [▶ 1450]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24 EnumValueCollection.T. Class

Collection of EnumValues [▶ 1443]

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.EnumValueCollection.T.
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class EnumValueCollection<T> : IList<EnumValue<T>>,
    ICollection<EnumValue<T>>, IEnumerable<EnumValue<T>>, IEnumerable
where T : IConvertible
```

VB

```
Public Class EnumValueCollection(Of T As IConvertible)
    Implements IList(Of EnumValue(Of T)), ICollection(Of EnumValue(Of T)),
    IEnumerable(Of EnumValue(Of T)), IEnumerable
```

Type Parameters

T	Base type of enum

The EnumValueCollection.T. type exposes the following members.

Properties

	Name	Description
	Count [▶ 1469]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1470]	Gets a value indicating whether the ICollection.T. is read-only.
	Item [▶ 1470]	Gets or sets the element at the specified index.

Methods

	Name	Description
	Add [▶ 1472]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1473]	Gets as read only.
	Clear [▶ 1473]	Removes all items from the ICollection.T..
	Contains(String) [▶ 1474]	Determines whether [contains] [the specified name].
	Contains(T) [▶ 1475]	Determines whether [contains] [the specified value].
	Contains(EnumValue.T.) [▶ 1475]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1476]	Copies the entire list.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1477]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetNames [▶ 1477]	Gets the names.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	GetValues [▶ 1478]	Gets the values.
	IndexOf [▶ 1478]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1479]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Parse [▶ 1479]	Parses the specified string to the Enum value.
	Remove [▶ 1480]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1481]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInfo [▶ 1482]	Tries the get information.
	TryParse [▶ 1482]	Parse the specified string to the enum value.

Operators

	Name	Description
	explicit (EnumValueCollection.T. to EnumValueCollection) [▶ 1483]	Performs an explicit conversion from EnumValueCollection.T. to EnumValueCollection [▶ 1450].

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.1 EnumValueCollection.T. Properties

The EnumValueCollection.T. [▶ 1467] generic type exposes the following members.

Properties

	Name	Description
	Count [▶ 1469]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1470]	Gets a value indicating whether the ICollection.T. is read-only.
	Item [▶ 1470]	Gets or sets the element at the specified index.

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.1.1 EnumValueCollection.T..Count Property

Gets the number of elements contained in the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Count { get; }
```

VB

```
Public ReadOnly Property Count As Integer
    Get
```

Property Value

Type: Int32
The count.

Implements

ICollection.T..Count

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.1.2 EnumValueCollection.T..IsReadOnly Property

Gets a value indicating whether the ICollection.T. is read-only.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is read only; otherwise, false.

Implements

ICollection.T..IsReadOnly

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.1.3 EnumValueCollection.T..Item Property

Gets or sets the element at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public EnumValue<T> this[
    int index
] { get; set; }
```

VB

```
Public Default Property Item (
    index As Integer
) As EnumValue(Of T)
    Get
    Set
```

Parameters

index	Type: System.Int32 The index.

Return Value

Type: EnumValue [▶ 1443].T [▶ 1467].
EnumValue<T>.

Implements

IList.T..Item.Int32.

Exceptions

Exception	Condition
NotImplementedException	

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2 EnumValueCollection.T. Methods

The EnumValueCollection.T. [▶ 1467] generic type exposes the following members.

Methods

	Name	Description
	Add [▶ 1472]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1473]	Gets as read only.
	Clear [▶ 1473]	Removes all items from the ICollection.T..
	Contains(String) [▶ 1474]	Determines whether [contains] [the specified name].
	Contains(T) [▶ 1475]	Determines whether [contains] [the specified value].
	Contains(EnumValue.T.) [▶ 1475]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1476]	Copies the entire list.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1477]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetNames [▶ 1477]	Gets the names.
	GetType	Gets the Type of the current instance. (Inherited from Object.)

	Name	Description
	GetValues [▶ 1478]	Gets the values.
	IndexOf [▶ 1478]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1479]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Parse [▶ 1479]	Parses the specified string to the Enum value.
	Remove [▶ 1480]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1481]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInfo [▶ 1482]	Tries the get information.
	TryParse [▶ 1482]	Parse the specified string to the enum value.

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.1 EnumValueCollection.T..Add Method

Adds an item to the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Add(
    EnumValue<T> item
)
```

VB

```
Public Sub Add (
    item As EnumValue(Of T)
)
```

Parameters

item	Type: TwinCAT.TypeSystem.EnumValue [▶ 1443].T [▶ 1467]. The object to add to the ICollection.T..

Implements

ICollection.T..Add(T)

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.2 EnumValueCollection.T..AsReadOnly Method

Gets as read only.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyEnumValueCollection<T> AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlyEnumValueCollection(Of T)
```

Field Value

Type: ReadOnlyEnumValueCollection [▶ 1743].T [▶ 1467].
As read only.

Return Value

Type: ReadOnlyEnumValueCollection [▶ 1743].T [▶ 1467].
ReadOnlyEnumValueCollection<T>.

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.3 EnumValueCollection.T..Clear Method

Removes all items from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Clear()
```

VB

```
Public Sub Clear
```

Implements

ICollection.T..Clear.

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.4 EnumValueCollection.T..Contains Method

Overload List

	Name	Description
	Contains(String) [▶ 1474]	Determines whether [contains] [the specified name].
	Contains(T) [▶ 1475]	Determines whether [contains] [the specified value].
	Contains(EnumValue.T.) [▶ 1475]	Determines whether the ICollection.T. contains a specific value.

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

EnumValueCollection.T..Contains Method (String)

Determines whether [contains] [the specified name].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    string name
)
```

VB

```
Public Function Contains (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name.

Return Value

Type: Boolean
true if [contains] [the specified name]; otherwise, false.

Reference

EnumValueCollection.T. Class [▶ 1467]

Contains Overload [▶ 1474]

TwinCAT.TypeSystem Namespace [▶ 1184]

EnumValueCollection.T..Contains Method (T)

Determines whether [contains] [the specified value].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    T value
)
```

VB

```
Public Function Contains (
    value As T
) As Boolean
```

Parameters

value	Type: T [▶ 1467] The value.

Return Value

Type: Boolean
true if [contains] [the specified value]; otherwise, false.

Reference

EnumValueCollection.T. Class [▶ 1467]

Contains Overload [▶ 1474]

TwinCAT.TypeSystem Namespace [▶ 1184]

EnumValueCollection.T..Contains Method (EnumValue.T.)

Determines whether the ICollection.T. contains a specific value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    EnumValue<T> item
)
```

VB

```
Public Function Contains (
    item As EnumValue(Of T)
) As Boolean
```

Parameters

item	Type: TwinCAT.TypeSystem.EnumValue [▶ 1443].T [▶ 1467]. The object to locate in the ICollection.T..

Return Value

Type: Boolean
true if item is found in the ICollection.T.; otherwise, false.

Implements

ICollection.T..Contains(T)

Reference

EnumValueCollection.T. Class [▶ 1467]

Contains Overload [▶ 1474]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.5 EnumValueCollection.T..CopyTo Method

Copies the entire list.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void CopyTo(
    EnumValue<T>[] array,
    int arrayIndex
)
```

VB

```
Public Sub CopyTo (
    array As EnumValue(Of T)(),
    arrayIndex As Integer
)
```

Parameters

array	Type: .TwinCAT.TypeSystem.EnumValue [▶ 1443].T [▶ 1467].. The array.
arrayIndex	Type: System.Int32 Index of the array.

Implements

ICollection.T..CopyTo(.T., Int32)

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.6 EnumValueCollection.T..GetEnumerator Method

Returns an enumerator that iterates through the collection.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumerator<EnumValue<T>> GetEnumerator()
```

VB

```
Public Function GetEnumerator As IEnumerator(Of EnumValue(Of T))
```

Return Value

Type: IEnumerator.EnumValue [▶ 1443].T [▶ 1467]..
A IEnumerator.T. that can be used to iterate through the collection.

Implements

IEnumerable.T..GetEnumerator.

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.7 EnumValueCollection.T..GetNames Method

Gets the names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string[] GetNames()
```

VB

```
Public Function GetNames As String()
```

Return Value

Type: .String.
System.String[].

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.8 EnumValueCollection.T..GetValues Method

Gets the values.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T[] GetValues()
```

VB

```
Public Function GetValues As T()
```

Return Value

Type: .T [▶ 1467].
T[].

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.9 EnumValueCollection.T..IndexOf Method

Determines the index of a specific item in the IList.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int IndexOf(
    EnumValue<T> item
)
```

VB

```
Public Function IndexOf (
    item As EnumValue(Of T)
) As Integer
```

Parameters

item	Type: TwinCAT.TypeSystem.EnumValue [▶ 1443].T [▶ 1467]. The object to locate in the IList.T..

Return Value

Type: Int32
The index of item if found in the list; otherwise, -1.

Implements

IList.T..IndexOf(T)

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.10 EnumValueCollection.T..Insert Method

Inserts an item to the IList.T. at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Insert(
    int index,
    EnumValue<T> item
)
```

VB

```
Public Sub Insert (
    index As Integer,
    item As EnumValue(Of T)
)
```

Parameters

index	Type: System.Int32 The zero-based index at which item should be inserted.
item	Type: TwinCAT.TypeSystem.EnumValue [▶ 1443].T [▶ 1467]. The object to insert into the IList.T..

Implements

IList.T..Insert(Int32, T)

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.11 EnumValueCollection.T..Parse Method

Parses the specified string to the Enum value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T Parse(
    string name
)
```

VB

```
Public Function Parse (
    name As String
) As T
```

Parameters

name	Type: System.String The name.

Return Value

Type: T [▶ 1467]
T.

Exceptions

Exception	Condition
ArgumentOutOfRangeException	name

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.12 EnumValueCollection.T..Remove Method

Removes the first occurrence of a specific object from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Remove(
    EnumValue<T> item
)
```

VB

```
Public Function Remove (
    item As EnumValue(Of T)
) As Boolean
```

Parameters

item	Type: TwinCAT.TypeSystem.EnumValue [▶ 1443].T [▶ 1467]. The object to remove from the ICollection.T..

Return Value

Type: Boolean
true if item was successfully removed from the ICollection.T.; otherwise, false. This method also returns false if item is not found in the original ICollection.T..

Implements

ICollection.T..Remove(T)

Exceptions

Exception	Condition
NotImplementedException	

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.13 EnumValueCollection.T..RemoveAt Method

Removes the IList.T. item at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void RemoveAt(
    int index
)
```

VB

```
Public Sub RemoveAt (
    index As Integer
)
```

Parameters

index	Type: System.Int32 The zero-based index of the item to remove.

Implements

IList.T..RemoveAt(Int32)

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.14 EnumValueCollection.T..TryGetInfo Method

Tries the get information.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetInfo(
    T val,
    out EnumValue<T> ei
)
```

VB

```
Public Function TryGetInfo (
    val As T,
    <OutAttribute> ByRef ei As EnumValue(Of T)
) As Boolean
```

Parameters

val	Type: T [▶ 1467] The value.
ei	Type: TwinCAT.TypeSystem.EnumValue [▶ 1443].T [▶ 1467].. The ei.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.2.15 EnumValueCollection.T..TryParse Method

Parse the specified string to the enum value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryParse(
    string name,
    out T value
)
```

VB

```
Public Function TryParse (
    name As String,
    <OutAttribute> ByRef value As T
) As Boolean
```

Parameters

name	Type: System.String The name.
value	Type: T [▶ 1467]. The value.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.3 EnumValueCollection.T. Type Conversions

The EnumValueCollection.T. [▶ 1467] generic type exposes the following members.

Operators

	Name	Description
	. (EnumValueCollection.T. to EnumValueCollection) [▶ 1483]	Performs an explicit conversion from EnumValueCollection.T. [▶ 1467] to EnumValueCollection [▶ 1450].

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.24.3.1 EnumValueCollection.T. . Conversion (EnumValueCollection.T. to EnumValueCollection)

Performs an explicit conversion from EnumValueCollection.T. [▶ 1467] to EnumValueCollection [▶ 1450].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static explicit operator EnumValueCollection (
    EnumValueCollection<T> coll
)
```

VB

```
Public Shared Narrowing Operator CType (
    coll As EnumValueCollection(Of T)
) As EnumValueCollection
```

Parameters

coll	Type: TwinCAT.TypeSystem.EnumValueCollection [▶ 1467].T [▶ 1467]. The coll.

Return Value

Type: EnumValueCollection [▶ 1450]
The result of the conversion.

Reference

EnumValueCollection.T. Class [▶ 1467]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.25 FieldCollection Class

Collection of IField [▶ 1554] objects.

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.Generic.InstanceCollection [▶ 1887].IField [▶ 1554].
    TwinCAT.TypeSystem.FieldCollection
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class FieldCollection : InstanceCollection<IField>
```

VB

```
Public Class FieldCollection
    Inherits InstanceCollection(Of IField)
```

The FieldCollection type exposes the following members.

Constructors

	Name	Description
	FieldCollection. [▶ 1486]	Initializes a new instance of the FieldCollection class.
	FieldCollection(IEnumerable.IField.) [▶ 1487]	Initializes a new instance of the FieldCollection class (copy constructor)

Properties

	Name	Description
	Count [▶ 1891]	Gets the collection count. (Inherited from InstanceCollection.T. [▶ 1887].)
	IsReadOnly [▶ 1892]	Gets a value indicating whether this instance is read only. (Inherited from InstanceCollection.T. [▶ 1887].)

	Name	Description
	Item.Int32. [▶ 1893]	Gets or sets the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.String. [▶ 1894]	Gets the IInstance [▶ 1556] with the specified instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	Mode [▶ 1894]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from InstanceCollection.T. [▶ 1887].)

Methods

	Name	Description
	Add [▶ 1896]	Adds the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	AddRange [▶ 1897]	Adds the specified items to this collection. (Inherited from InstanceCollection.T. [▶ 1887].)
	AsReadOnly [▶ 1489]	Returns a read only copy of this collection (shallow copy)
	Clear [▶ 1898]	Clears this instance. (Inherited from InstanceCollection.T. [▶ 1887].)
	Clone [▶ 1489]	Clones this FieldCollection.
	Contains(String) [▶ 1899]	Determines whether this collection contains an IInstance [▶ 1556] with the specified InstanceName / InstancePath (Inherited from InstanceCollection.T. [▶ 1887].)
	Contains(T) [▶ 1899]	Determines whether this collection contains the specified IInstance [▶ 1556] (Inherited from InstanceCollection.T. [▶ 1887].)
	ContainsName [▶ 1900]	Determines whether the specified instance name contains name. (Inherited from InstanceCollection.T. [▶ 1887].)
	CopyTo [▶ 1901]	Copies this InstanceCollection.T. [▶ 1887] to the specified array. (Inherited from InstanceCollection.T. [▶ 1887].)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1902]	Gets the enumerator. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1902]	Gets the IInstance [▶ 1556]by instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetInstanceByName [▶ 1903]	Gets the name of the instance by. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1904]	Determines the index of the specified IInstance [▶ 1556]. (Inherited from InstanceCollection.T. [▶ 1887].)
	Insert [▶ 1905]	Inserts the specified IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1905]	Removes the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	RemoveAt [▶ 1906]	Removes the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)

	Name	Description
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1907]	Tries to get the IInstance [▶ 1556]. of the specified path. (Inherited from InstanceCollection.T. [▶ 1887].)
	TryGetInstanceByName [▶ 1908]	Tries to get Instnaces by name. (Inherited from InstanceCollection.T. [▶ 1887].)

Fields

	Name	Description
	_list [▶ 1909]	The _list (Inherited from InstanceCollection.T. [▶ 1887].)
	_pathDict [▶ 1909]	The _path dictionary (Inherited from InstanceCollection.T. [▶ 1887].)
	mode [▶ 1910]	The mode this InstanceCollection.T. [▶ 1887] is working in. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.25.1 FieldCollection Constructor

Overload List

	Name	Description
	FieldCollection. [▶ 1486]	Initializes a new instance of the FieldCollection [▶ 1484] class.
	FieldCollection(IEnumerable.IField.) [▶ 1487]	Initializes a new instance of the FieldCollection [▶ 1484] class (copy constructor)

Reference

FieldCollection Class [▶ 1484]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.25.1.1 FieldCollection Constructor

Initializes a new instance of the FieldCollection [▶ 1484] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public FieldCollection()
```

VB

```
Public Sub New
```

Reference

FieldCollection Class [▶ 1484]

FieldCollection Overload [▶ 1486]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.25.1.2 FieldCollection Constructor (IEnumerable.IField.)

Initializes a new instance of the FieldCollection [▶ 1484] class (copy constructor)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public FieldCollection(
    IEnumerable<IField> coll
)
```

VB

```
Public Sub New (
    coll As IEnumerable(Of IField)
)
```

Parameters

coll	Type: System.Collections.Generic.IEnumerable.IField [▶ 1554]. The coll.

Reference

FieldCollection Class [▶ 1484]

FieldCollection Overload [▶ 1486]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.25.2 FieldCollection Properties

The FieldCollection [▶ 1484] type exposes the following members.

Properties

	Name	Description
	Count [▶ 1891]	Gets the collection count. (Inherited from InstanceCollection.T. [▶ 1887].)
	IsReadOnly [▶ 1892]	Gets a value indicating whether this instance is read only. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.Int32. [▶ 1893]	Gets or sets the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)

	Name	Description
	Item.String. [▶ 1894]	Gets the IInstance [▶ 1556] with the specified instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	Mode [▶ 1894]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

FieldCollection Class [▶ 1484]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.25.3 FieldCollection Methods

The FieldCollection [▶ 1484] type exposes the following members.

Methods

	Name	Description
	Add [▶ 1896]	Adds the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	AddRange [▶ 1897]	Adds the specified items to this collection. (Inherited from InstanceCollection.T. [▶ 1887].)
	AsReadOnly [▶ 1489]	Returns a read only copy of this collection (shallow copy)
	Clear [▶ 1898]	Clears this instance. (Inherited from InstanceCollection.T. [▶ 1887].)
	Clone [▶ 1489]	Clones this FieldCollection [▶ 1484].
	Contains(String) [▶ 1899]	Determines whether this collection contains an IInstance [▶ 1556] with the specified InstanceName / InstancePath (Inherited from InstanceCollection.T. [▶ 1887].)
	Contains(T) [▶ 1899]	Determines whether this collection contains the specified IInstance [▶ 1556] (Inherited from InstanceCollection.T. [▶ 1887].)
	ContainsName [▶ 1900]	Determines whether the specified instance name contains name. (Inherited from InstanceCollection.T. [▶ 1887].)
	CopyTo [▶ 1901]	Copies this InstanceCollection.T. [▶ 1887] to the specified array. (Inherited from InstanceCollection.T. [▶ 1887].)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1902]	Gets the enumerator. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1902]	Gets the IInstance [▶ 1556]by instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetInstanceByName [▶ 1903]	Gets the name of the instance by. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1904]	Determines the index of the specified IInstance [▶ 1556]. (Inherited from InstanceCollection.T. [▶ 1887].)

	Name	Description
	Insert [▶ 1905]	Inserts the specified IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1905]	Removes the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	RemoveAt [▶ 1906]	Removes the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1907]	Tries to get the IInstance [▶ 1556]. of the specified path. (Inherited from InstanceCollection.T. [▶ 1887].)
	TryGetInstanceByName [▶ 1908]	Tries to get Instnaces by name. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

FieldCollection Class [▶ 1484]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.25.3.1 FieldCollection.AsReadOnly Method

Returns a read only copy of this collection (shallow copy)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyFieldCollection AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlyFieldCollection
```

Return Value

Type: ReadOnlyFieldCollection [▶ 1750]
The readonly copy.

Reference

FieldCollection Class [▶ 1484]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.25.3.2 FieldCollection.Clone Method

Clones this FieldCollection [▶ 1484].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public FieldCollection Clone()
```

VB

```
Public Function Clone As FieldCollection
```

Return Value

Type: FieldCollection [▶ 1484]
A cloned FieldCollection [▶ 1484].

Reference

FieldCollection Class [▶ 1484]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.25.4 FieldCollection Fields

The FieldCollection [▶ 1484] type exposes the following members.

Fields

	Name	Description
	_list [▶ 1909]	The _list (Inherited from InstanceCollection.T. [▶ 1887].)
	_pathDict [▶ 1909]	The _path dictionary (Inherited from InstanceCollection.T. [▶ 1887].)
	mode [▶ 1910]	The mode this InstanceCollection.T. [▶ 1887] is working in. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

FieldCollection Class [▶ 1484]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.26 IAliasInstance Interface

Interface representing an instance of an IAliasType [▶ 1493].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IAliasInstance : ISymbol,
    IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IAliasInstance
    Inherits ISymbol, IAttributedInstance, IInstance, IBitSize
```

The IAliasInstance type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.ISymbol [▶ 1634]

5.8.26.1 IAliasInstance Properties

The IAliasInstance [▶ 1490] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)

	Name	Description
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IAliasInstance Interface [▶ 1490]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.27 IAliasType Interface

Interface representing an Alias Type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IAliasType : IDataType,
    IBitSize
```

VB

```
Public Interface IAliasType
    Inherits IDataType, IBitSize
```

The IAliasType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 1495]	Gets the Base Type
	BaseTypeName [▶ 1496]	Gets the BaseType name
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)

	Name	Description
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.27.1 IAliasType Properties

The IAliasType [▶ 1493] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 1495]	Gets the Base Type
	BaseTypeName [▶ 1496]	Gets the BaseType name
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)

	Name	Description
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

IAliasType Interface [▶ 1493]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.27.1.1 IAliasType.BaseType Property

Gets the Base Type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType BaseType { get; }
```

VB

```
ReadOnly Property BaseType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]

Reference

IAliasType Interface [▶ 1493]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.27.1.2 IAliasType.BaseTypeName Property

Gets the BaseType name

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string BaseTypeName { get; }
```

VB

```
ReadOnly Property BaseTypeName As String
    Get
```

Property Value

Type: String

Reference

IAliasType Interface [▶ 1493]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.28 IArrayInstance Interface

Interface representing an array instance

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IArrayInstance : ISymbol,
    IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IArrayInstance
    Inherits ISymbol, IAttributedInstance, IInstance, IBitSize
```

The IArrayInstance type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)

	Name	Description
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	Dimensions [▶ 1499]	Gets the dimensions as read only collection.
	Elements [▶ 1500]	Gets the contained Array Elements as read only collection.
	ElementType [▶ 1500]	Gets the type of the contained elements.
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Item [▶ 1501]	Gets the ISymbol [▶ 1634] with the specified indices.
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Methods

	Name	Description
	TryGetElement(IList.., Int32.., ISymbol.) [▶ 1502]	Tries to get the array element with the specified indices (jagged array support).
	TryGetElement(.Int32., ISymbol.) [▶ 1503]	Tries to get the array element with specified indices (only first level on jagged arrays)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.28.1 IArrayInstance Properties

The IArrayInstance [▶ 1496] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	Dimensions [▶ 1499]	Gets the dimensions as read only collection.
	Elements [▶ 1500]	Gets the contained Array Elements as read only collection.
	ElementType [▶ 1500]	Gets the type of the contained elements.
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)

	Name	Description
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Item [▶ 1501]	Gets the ISymbol [▶ 1634] with the specified indices.
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IArrayInstance Interface [▶ 1496]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.28.1.1 IArrayInstance.Dimensions Property

Gets the dimensions as read only collection.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyDimensionCollection Dimensions { get; }
```

VB

```
ReadOnly Property Dimensions As ReadOnlyDimensionCollection
    Get
```

Property Value

Type: ReadOnlyDimensionCollection [▶ 1731]
The dimensions.

Reference

IArrayInstance Interface [▶ 1496]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.28.1.2 IArrayInstance.Elements Property

Gets the contained Array Elements as read only collection.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlySymbolCollection Elements { get; }
```

VB

```
ReadOnly Property Elements As ReadOnlySymbolCollection
    Get
```

Property Value

Type: ReadOnlySymbolCollection [▶ 1772]
The elements.

Reference

IArrayInstance Interface [▶ 1496]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.28.1.3 IArrayInstance.ElementType Property

Gets the type of the contained elements.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType ElementType { get; }
```

VB

```
ReadOnly Property ElementType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the element.

Reference

IArrayInstance Interface [▶ 1496]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.28.1.4 IArrayInstance.Item Property

Gets the ISymbol [▶ 1634] with the specified indices.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ISymbol this[
    int[] indices
] { get; }
```

VB

```
ReadOnly Default Property Item (
    indices As Integer()
) As ISymbol
    Get
```

Parameters

indices	Type: .System.Int32. The indices.

Property Value

Type: ISymbol [▶ 1634]
The ISymbol [▶ 1634].

Return Value

Type: ISymbol [▶ 1634]

Reference

IArrayInstance Interface [▶ 1496]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.28.2 IArrayInstance Methods

Methods

	Name	Description
	TryGetElement(IList.. Int32.., ISymbol.) [▶ 1502]	Tries to get the array element with the specified indices (jagged array support).
	TryGetElement(.Int32., ISymbol.) [▶ 1503]	Tries to get the array element with specified indices (only first level on jagged arrays)

Reference

IArrayInstance Interface [▶ 1496]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.28.2.1 IArrayInstance.TryGetElement Method

Overload List

	Name	Description
	TryGetElement(IList..Int32.., ISymbol.) [▶ 1502]	Tries to get the array element with the specified indices (jagged array support).
	TryGetElement(.Int32., ISymbol.) [▶ 1503]	Tries to get the array element with specified indices (only first level on jagged arrays)

Reference

IArrayInstance Interface [▶ 1496]

TwinCAT.TypeSystem Namespace [▶ 1184]

IArrayInstance.TryGetElement Method (IList..Int32.., ISymbol.)

Tries to get the array element with the specified indices (jagged array support).

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TryGetElement(
    IList<int[]> jaggedIndices,
    out ISymbol symbol
)
```

VB

```
Function TryGetElement (
    jaggedIndices As IList(Of Integer()),
    <OutAttribute> ByRef symbol As ISymbol
) As Boolean
```

Parameters

jaggedIndices	Type: System.Collections.Generic.IList..Int32.. The jagged indices list.
symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634]. The symbol.

Return Value

Type: Boolean
true if found, false if the jagged indices specifiers is out-of-range.

Reference

IArrayInstance Interface [▶ 1496]

TryGetElement Overload [▶ 1502]

TwinCAT.TypeSystem Namespace [▶ 1184]

IArrayInstance.TryGetElement Method (.Int32., ISymbol.)

Tries to get the array element with specified indices (only first level on jagged arrays)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TryGetElement(
    int[] indices,
    out ISymbol symbol
)
```

VB

```
Function TryGetElement (
    indices As Integer(),
    <OutAttribute> ByRef symbol As ISymbol
) As Boolean
```

Parameters

indices	Type: .System.Int32. The indices.
symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634]. The found Array element symbol (out-parameter).

Return Value

Type: Boolean
true if found, false if the indices specifiers is out-of-range.

Reference

IArrayInstance Interface [▶ 1496]

TryGetElement Overload [▶ 1502]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.29 IArrayType Interface

Interface representing an array DataType [▶ 1517].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IArrayType : IDataType,
    IBitSize
```

VB

```
Public Interface IArrayType
    Inherits IDataType, IBitSize
```

The IArrayType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	Dimensions [▶ 1506]	Gets the dimensions as read only collection.
	ElementType [▶ 1506]	Gets the type of the contained elements.
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsJagged [▶ 1507]	Gets a value indicating whether this instance is jagged.
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	JaggedLevel [▶ 1507]	Gets the jagged level (Non-Jagged Array have level 1)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.29.1 IArrayType Properties

The IArrayType [▶ 1503] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	Dimensions [▶ 1506]	Gets the dimensions as read only collection.
	ElementType [▶ 1506]	Gets the type of the contained elements.
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsJagged [▶ 1507]	Gets a value indicating whether this instance is jagged.
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	JaggedLevel [▶ 1507]	Gets the jagged level (Non-Jagged Array have level 1)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

IArrayType Interface [▶ 1503]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.29.1.1 IArrayType.Dimensions Property

Gets the dimensions as read only collection.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyDimensionCollection Dimensions { get; }
```

VB

```
ReadOnly Property Dimensions As ReadOnlyDimensionCollection
    Get
```

Property Value

Type: ReadOnlyDimensionCollection [▶ 1731]
The dimensions.

Reference

IArrayType Interface [▶ 1503]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.29.1.2 IArrayType.ElementType Property

Gets the type of the contained elements.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType ElementType { get; }
```

VB

```
ReadOnly Property ElementType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the element.

Reference

IArrayType Interface [▶ 1503]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.29.1.3 IArrayType.IsJagged Property

Gets a value indicating whether this instance is jagged.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsJagged { get; }
```

VB

```
ReadOnly Property IsJagged As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is jagged; otherwise, false.

Reference

IArrayType Interface [▶ 1503]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.29.1.4 IArrayType.JaggedLevel Property

Gets the jagged level (Non-Jagged Array have level 1)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int JaggedLevel { get; }
```

VB

```
ReadOnly Property JaggedLevel As Integer
    Get
```

Property Value

Type: Int32
The jagged level.

Reference

IArrayType Interface [▶ 1503]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.30 IArrayValue Interface

Interface IArrayValue

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IArrayValue : IValue
```

VB

```
Public Interface IArrayValue
    Inherits IValue
```

The IArrayValue type exposes the following members.

Properties

	Name	Description
	Age [▶ 1660]	Gets the age of the value (last successful read of the value) (Inherited from IValue [▶ 1659].)
	CachedRaw [▶ 1661]	Gets the cached Raw internal Data. (Inherited from IValue [▶ 1659].)
	DataType [▶ 1661]	Gets the data type bound to this IValue [▶ 1659] (Inherited from IValue [▶ 1659].)
	IsPrimitive [▶ 1662]	Gets a value indicating whether this IValue [▶ 1659] is a primitive value. (Inherited from IValue [▶ 1659].)
	Symbol [▶ 1662]	Gets the symbol bound to this IValue [▶ 1659]. (Inherited from IValue [▶ 1659].)
	UtcTimeStamp [▶ 1663]	Gets the Time stamp of the last successful read of the Value (local user time, UTC) (Inherited from IValue [▶ 1659].)

Methods

	Name	Description
	Read [▶ 1664]	Reads the value (via ADS) (Inherited from IValue [▶ 1659].)
	ResolveValue [▶ 1664]	Resolves the Value object to its primitive value. (Inherited from IValue [▶ 1659].)
	TryGetArrayElementValues [▶ 1510]	Returns Array Element values.
	TryGetIndexValue [▶ 1510]	Tries to get the specified Array Element
	TryResolveValue [▶ 1665]	Tries to resolves the Value object to its primitive value. (Inherited from IValue [▶ 1659].)
	TrySetIndexValue [▶ 1511]	Tries to set the indexed value on Arrays
	Write [▶ 1665]	Writes the value (via ADS) (Inherited from IValue [▶ 1659].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IValue [▶ 1659]

5.8.30.1 IArrayValue Properties

The IArrayValue [▶ 1507] type exposes the following members.

Properties

	Name	Description
	Age [▶ 1660]	Gets the age of the value (last successful read of the value) (Inherited from IValue [▶ 1659].)
	CachedRaw [▶ 1661]	Gets the cached Raw internal Data. (Inherited from IValue [▶ 1659].)
	DataType [▶ 1661]	Gets the data type bound to this IValue [▶ 1659] (Inherited from IValue [▶ 1659].)
	IsPrimitive [▶ 1662]	Gets a value indicating whether this IValue [▶ 1659] is a primitive value. (Inherited from IValue [▶ 1659].)
	Symbol [▶ 1662]	Gets the symbol bound to this IValue [▶ 1659]. (Inherited from IValue [▶ 1659].)
	UtcTimeStamp [▶ 1663]	Gets the Time stamp of the last successful read of the Value (local user time, UTC) (Inherited from IValue [▶ 1659].)

Reference

IArrayValue Interface [▶ 1507]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.30.2 IArrayValue Methods

The IArrayValue [▶ 1507] type exposes the following members.

Methods

	Name	Description
	Read [▶ 1664]	Reads the value (via ADS) (Inherited from IValue [▶ 1659].)
	ResolveValue [▶ 1664]	Resolves the Value object to its primitive value. (Inherited from IValue [▶ 1659].)
	TryGetArrayElementValues [▶ 1510]	Returns Array Element values.
	TryGetIndexValue [▶ 1510]	Tries to get the specified Array Element
	TryResolveValue [▶ 1665]	Tries to resolves the Value object to its primitive value. (Inherited from IValue [▶ 1659].)
	TrySetIndexValue [▶ 1511]	Tries to set the indexed value on Arrays
	Write [▶ 1665]	Writes the value (via ADS) (Inherited from IValue [▶ 1659].)

Reference

IArrayValue Interface [▶ 1507]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.30.2.1 IArrayValue.TryGetArrayElementValues Method

Returns Array Element values.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TryGetArrayElementValues(
    out IEnumerable<Object> elementValues
)
```

VB

```
Function TryGetArrayElementValues (
    <OutAttribute> ByRef elementValues As IEnumerable(Of Object)
) As Boolean
```

Parameters

elementValues	Type: System.Collections.Generic.IEnumerable.Object.. The element values.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

IArrayValue Interface [▶ 1507]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.30.2.2 IArrayValue.TryGetIndexValue Method

Tries to get the specified Array Element

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TryGetIndexValue(
    int[] indices,
    out Object value
)
```

VB

```
Function TryGetIndexValue (
    indices As Integer(),
    <OutAttribute> ByRef value As Object
) As Boolean
```

Parameters

indices	Type: .System.Int32. The indices.
value	Type: System.Object. The value.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

IArrayValue Interface [▶ 1507]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.30.2.3 IArrayValue.TrySetIndexValue Method

Tries to set the indexed value on Arrays

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TrySetIndexValue(
    Object[] indexes,
    Object value
)
```

VB

```
Function TrySetIndexValue (
    indexes As Object(),
    value As Object
) As Boolean
```

Parameters

indexes	Type: .System.Object. The indexes.
value	Type: System.Object The value.

Return Value

Type: Boolean
true if succeeded, false otherwise.

Reference

IArrayValue Interface [▶ 1507]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.31 IAttributedInstance Interface

Interface IAttributedInstance

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IAttributedInstance : IInstance,
    IBitSize
```

VB

```
Public Interface IAttributedInstance
    Inherits IInstance, IBitSize
```

The IAttributedInstance type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes.
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IInstance [▶ 1556]

5.8.31.1 IAttributedInstance Properties

The IAttributedInstance [▶ 1512] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes.
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IAttributedInstance Interface [▶ 1512]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.31.1.1 IAttributedInstance.Attributes Property

Gets the Type Attributes.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyTypeAttributeCollection Attributes { get; }
```

VB

```
ReadOnly Property Attributes As ReadOnlyTypeAttributeCollection
    Get
```

Property Value

Type: ReadOnlyTypeAttributeCollection [▶ 1776]
The attributes.

Reference

IAttributedInstance Interface [▶ 1512]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.32 IBitSize Interface

Interface IBitSize

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IBitSize
```

VB

```
Public Interface IBitSize
```

The IBitSize type exposes the following members.

Properties

	Name	Description
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits.
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516]

Remarks

Specifies the Bitness of the the object and the bit resp. byte sizes.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.32.1 IBitSize Properties

The IBitSize [▶ 1514] type exposes the following members.

Properties

	Name	Description
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits.
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516]

Reference

IBitSize Interface [▶ 1514]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.32.1.1 IBitSize.BitSize Property

Gets the size of the IDataType [▶ 1517] in bits.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int BitSize { get; }
```

VB

```
ReadOnly Property BitSize As Integer
    Get
```

Property Value

Type: Int32
The size of IDataType [▶ 1517] in bits.

Reference

IBitSize Interface [▶ 1514]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.32.1.2 IBitSize.ByteSize Property

Gets the (aligned) size of of the Type/Instance in Bytes

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int ByteSize { get; }
```

VB

```
ReadOnly Property ByteSize As Integer
    Get
```

Property Value

Type: Int32
The size of the byte.

Reference

IBitSize Interface [▶ 1514]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.32.1.3 IBitSize.IsBitType Property

Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsBitType { get; }
```

VB

```
ReadOnly Property IsBitType As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is bit mapping; otherwise, false.

Reference

IBitSize Interface [▶ 1514]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.32.1.4 IBitSize.IsByteAligned Property

Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsByteAligned { get; }
```

VB

```
ReadOnly Property IsByteAligned As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is byte aligned; otherwise, false.

Reference

IBitSize Interface [▶ 1514]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.32.1.5 IBitSize.Size Property

Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Size { get; }
```

VB

```
ReadOnly Property Size As Integer
    Get
```

Property Value

Type: Int32
The size of the bit.

Reference

IBitSize Interface [▶ 1514]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.33 IDataType Interface

Base interface for objects representing data types

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IDataType : IBitSize
```

VB

```
Public Interface IDataType
    Inherits IBitSize
```

The IDataType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType
	BitSize [▶ 1515]	Gets the size of the IDataType in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category
	Comment [▶ 1520]	Gets the comment behind the variable declaration.
	FullName [▶ 1521]	Gets the full name of the IDataType (Namespace + Name)
	Id [▶ 1521]	Gets the ID of the DataType
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType is a container type
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType is a pointer type
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType is primitive
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType is a reference type
	Name [▶ 1524]	Gets the name of the Data Type (without namespace)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType exists.
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.33.1 IDataType Properties

The IDataType [▶ 1517] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517]
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category
	Comment [▶ 1520]	Gets the comment behind the variable declaration.
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name)
	Id [▶ 1521]	Gets the ID of the DataType
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type
	Name [▶ 1524]	Gets the name of the Data Type (without namespace)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists.
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

IDataType Interface [▶ 1517]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.33.1.1 IDataType.Attributes Property

Gets the attributes of the IDataType [▶ 1517]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyTypeAttributeCollection Attributes { get; }
```

VB

```
ReadOnly Property Attributes As ReadOnlyTypeAttributeCollection
    Get
```

Property Value

Type: ReadOnlyTypeAttributeCollection [▶ 1776]
The attributes.

Reference

IDataType Interface [▶ 1517]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.33.1.2 IDataType.Category Property

Gets the Data Type category

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
DataTypeCategory Category { get; }
```

VB

```
ReadOnly Property Category As DataTypeCategory
    Get
```

Property Value

Type: DataTypeCategory [▶ 1189]
The category.

Reference

IDataType Interface [▶ 1517]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.33.1.3 IDataType.Comment Property

Gets the comment behind the variable declaration.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Comment { get; }
```

VB

```
ReadOnly Property Comment As String
    Get
```

Property Value

Type: String
Comment behind the variable declaration.

Reference

IDataType Interface [▶ 1517]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.33.1.4 IDataType.FullName Property

Gets the full name of the IDataType [▶ 1517] (Namespace + Name)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string FullName { get; }
```

VB

```
ReadOnly Property FullName As String
    Get
```

Property Value

Type: String
The full name.

Reference

IDataType Interface [▶ 1517]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.33.1.5 IDataType.Id Property

Gets the ID of the DataType

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Id { get; }
```

VB

```
ReadOnly Property Id As Integer
    Get
```

Property Value

Type: Int32
The id.

Reference

IDataType Interface [▶ 1517]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.33.1.6 IDataType.IsContainer Property

Gets a value indicating whether this IDataType [▶ 1517] is a container type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsContainer { get; }
```

VB

```
ReadOnly Property IsContainer As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is container type; otherwise, false.

Remarks

Container Types are all types that contain SubElements like

- Array [▶ 1189]
- Pointer [▶ 1189]
- Union [▶ 1189]
- Struct [▶ 1189]
- Function [▶ 1189]
- FunctionBlock [▶ 1189]
- Program [▶ 1189]

and the Alias [▶ 1189] and Reference [▶ 1189] types, if they have a container type as base type.

Reference

IDataType Interface [▶ 1517]

TwinCAT.TypeSystem Namespace [▶ 1184]

IDataType.Category [▶ 1520]

5.8.33.1.7 IDataType.IsPointer Property

Gets a value indicating whether this IDataType [▶ 1517] is a pointer type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsPointer { get; }
```

VB

```
ReadOnly Property IsPointer As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is pointer type; otherwise, false.

Remarks

Pointer types can be dereferenced with the '^' operator.

Reference

IDataType Interface [▶ 1517]

TwinCAT.TypeSystem Namespace [▶ 1184]

IDataType.Category [▶ 1520]

5.8.33.1.8 IDataType.IsPrimitive Property

Gets a value indicating whether this IDataType [▶ 1517] is primitive

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsPrimitive { get; }
```

VB

```
ReadOnly Property IsPrimitive As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is primitive; otherwise, false.

Reference

IDataType Interface [▶ 1517]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.33.1.9 IDataType.IsReference Property

Gets a value indicating whether this IDataType [▶ 1517] is a reference type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsReference { get; }
```

VB

```
ReadOnly Property IsReference As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is container type; otherwise, false.

Remarks

Reference types can be dereferenced.

Reference

IDataType Interface [▶ 1517]

TwinCAT.TypeSystem Namespace [▶ 1184]

IDataType.Category [▶ 1520]

5.8.33.1.10 IDataType.Name Property

Gets the name of the Data Type (without namespace)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Name { get; }
```

VB

```
ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Reference

IDataType Interface [▶ 1517]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.33.1.11 IDataType.Namespace Property

Gets the namespace string within the IDataType [▶ 1517] exists.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Namespace { get; }
```

VB

```
ReadOnly Property Namespace As String
    Get
```

Property Value

Type: String
The namespace.

Reference

IDataType Interface [▶ 1517]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.34 IDimension Interface

Interface representing a single Dimension of an ArrayType [▶ 1503].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IDimension
```

VB

```
Public Interface IDimension
```

The IDimension type exposes the following members.

Properties

	Name	Description
	ElementCount [▶ 1526]	Gets the number of elements within that IDimension.
	LowerBound [▶ 1527]	Gets the lower bound of elements within that IDimension.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.34.1 IDimension Properties

The IDimension [▶ 1525] type exposes the following members.

Properties

	Name	Description
	ElementCount [▶ 1526]	Gets the number of elements within that IDimension [▶ 1525].
	LowerBound [▶ 1527]	Gets the lower bound of elements within that IDimension [▶ 1525].

Reference

IDimension Interface [▶ 1525]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.34.1.1 IDimension.ElementCount Property

Gets the number of elements within that IDimension [▶ 1525].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int ElementCount { get; }
```

VB

```
ReadOnly Property ElementCount As Integer
    Get
```

Property Value

Type: Int32
The element count.

Reference

IDimension Interface [▶ 1525]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.34.1.2 IDimension.LowerBound Property

Gets the lower bound of elements within that IDimension [▶ 1525].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int LowerBound { get; }
```

VB

```
ReadOnly Property LowerBound As Integer
    Get
```

Property Value

Type: Int32
The lower bound.

Reference

IDimension Interface [▶ 1525]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.35 IDimensionCollection Interface

Interface IDimensionCollection

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IDimensionCollection : IList<IDimension>,
    ICollection<IDimension>, IEnumerable<IDimension>, IEnumerable
```

VB

```
Public Interface IDimensionCollection
    Inherits IList(Of IDimension), ICollection(Of IDimension),
    IEnumerable(Of IDimension), IEnumerable
```

The IDimensionCollection type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ICollection.T.. (Inherited from ICollection.IDimension [▶ 1525]..)
	ElementCount [▶ 1529]	Gets the Number of elements in all Dimensions

	Name	Description
	IsReadOnly	Gets a value indicating whether the ICollection.T. is read-only. (Inherited from ICollection.IDimension [▶ 1525]..)
	Item	Gets or sets the element at the specified index. (Inherited from IList.IDimension [▶ 1525]..)
	LowerBounds [▶ 1529]	Gets the lower bounds.
	UpperBounds [▶ 1530]	Gets the lower bounds.

Methods

	Name	Description
	Add	Adds an item to the ICollection.T.. (Inherited from ICollection.IDimension [▶ 1525]..)
	Clear	Removes all items from the ICollection.T.. (Inherited from ICollection.IDimension [▶ 1525]..)
	Contains	Determines whether the ICollection.T. contains a specific value. (Inherited from ICollection.IDimension [▶ 1525]..)
	CopyTo	Copies the elements of the ICollection.T. to an Array, starting at a particular Array index. (Inherited from ICollection.IDimension [▶ 1525]..)
	GetDimensionLengths [▶ 1531]	Gets an array the specifies the Lengths of each Array Dimension
	GetEnumerator	Returns an enumerator that iterates through the collection. (Inherited from IEnumerable.IDimension [▶ 1525]..)
	IndexOf	Determines the index of a specific item in the IList.T.. (Inherited from IList.IDimension [▶ 1525]..)
	Insert	Inserts an item to the IList.T. at the specified index. (Inherited from IList.IDimension [▶ 1525]..)
	Remove	Removes the first occurrence of a specific object from the ICollection.T.. (Inherited from ICollection.IDimension [▶ 1525]..)
	RemoveAt	Removes the IList.T. item at the specified index. (Inherited from IList.IDimension [▶ 1525]..)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IDimension [▶ 1525]

5.8.35.1 IDimensionCollection Properties

The IDimensionCollection [▶ 1527] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ICollection.T.. (Inherited from ICollection.IDimension [▶ 1525]..)

	Name	Description
	ElementCount [▶ 1529]	Gets the Number of elements in all Dimensions
	IsReadOnly	Gets a value indicating whether the ICollection.T. is read-only. (Inherited from ICollection.IDimension [▶ 1525]..)
	Item	Gets or sets the element at the specified index. (Inherited from IList.IDimension [▶ 1525]..)
	LowerBounds [▶ 1529]	Gets the lower bounds.
	UpperBounds [▶ 1530]	Gets the lower bounds.

Reference

IDimensionCollection Interface [▶ 1527]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.35.1.1 IDimensionCollection.ElementCount Property

Gets the Number of elements in all Dimensions

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int ElementCount { get; }
```

VB

```
ReadOnly Property ElementCount As Integer
    Get
```

Property Value

Type: Int32

Reference

IDimensionCollection Interface [▶ 1527]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.35.1.2 IDimensionCollection.LowerBounds Property

Gets the lower bounds.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int[] LowerBounds { get; }
```

VB

```
ReadOnly Property LowerBounds As Integer()
    Get
```

Property Value

Type: .Int32.
The lower bounds.

Reference

IDimensionCollection Interface [▶ 1527]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.35.1.3 IDimensionCollection.UpperBounds Property

Gets the lower bounds.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int[] UpperBounds { get; }
```

VB

```
ReadOnly Property UpperBounds As Integer()
    Get
```

Property Value

Type: .Int32.
The lower bounds.

Reference

IDimensionCollection Interface [▶ 1527]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.35.2 IDimensionCollection Methods

The IDimensionCollection [▶ 1527] type exposes the following members.

Methods

	Name	Description
	Add	Adds an item to the ICollection.T.. (Inherited from ICollection.IDimension [▶ 1525]..)
	Clear	Removes all items from the ICollection.T.. (Inherited from ICollection.IDimension [▶ 1525]..)
	Contains	Determines whether the ICollection.T. contains a specific value. (Inherited from ICollection.IDimension [▶ 1525]..)

	Name	Description
	CopyTo	Copies the elements of the ICollection.T. to an Array, starting at a particular Array index. (Inherited from ICollection.IDimension [▶ 1525]..)
	GetDimensionLengths [▶ 1531]	Gets an array the specifies the Lengths of each Array Dimension
	GetEnumerator	Returns an enumerator that iterates through the collection. (Inherited from IEnumerable.IDimension [▶ 1525]..)
	IndexOf	Determines the index of a specific item in the IList.T.. (Inherited from IList.IDimension [▶ 1525]..)
	Insert	Inserts an item to the IList.T. at the specified index. (Inherited from IList.IDimension [▶ 1525]..)
	Remove	Removes the first occurrence of a specific object from the ICollection.T.. (Inherited from ICollection.IDimension [▶ 1525]..)
	RemoveAt	Removes the IList.T. item at the specified index. (Inherited from IList.IDimension [▶ 1525]..)

Reference

IDimensionCollection Interface [▶ 1527]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.35.2.1 IDimensionCollection.GetDimensionLengths Method

Gets an array the specifies the Lengths of each Array Dimension

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int[] GetDimensionLengths()
```

VB

```
Function GetDimensionLengths As Integer()
```

Return Value

Type: .Int32.
System.Int32[].

Reference

IDimensionCollection Interface [▶ 1527]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.36 IDynamicSymbol Interface

Interface IDynamicSymbol

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IDynamicSymbol : ISymbol,
    IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IDynamicSymbol
    Inherits ISymbol, IAttributedInstance, IInstance, IBitSize
```

The IDynamicSymbol type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)

	Name	Description
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	NormalizedName [▶ 1534]	Gets the normalized instance name (fixed name for dynamic property access)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.36.1 IDynamicSymbol Properties

The IDynamicSymbol [▶ 1531] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)

	Name	Description
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	NormalizedName [▶ 1534]	Gets the normalized instance name (fixed name for dynamic property access)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IDynamicSymbol Interface [▶ 1531]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.36.1.1 IDynamicSymbol.NormalizedName Property

Gets the normalized instance name (fixed name for dynamic property access)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string NormalizedName { get; }
```

VB

```
ReadOnly Property NormalizedName As String
    Get
```

Property Value

Type: String
The normalized instance name (can be the same like InstanceName [▶ 1559]

Reference

IDynamicSymbol Interface [▶ 1531]

TwinCAT.TypeSystem Namespace [▶ 1184]

IInstance.InstanceName [▶ 1559]

ISymbolFactory.InvalidCharacters

5.8.37 IDynamicSymbolLoader Interface

Dynamic symbol loader interface

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IDynamicSymbolLoader : ISymbolLoader,
    ISymbolProvider
```

VB

```
Public Interface IDynamicSymbolLoader
    Inherits ISymbolLoader, ISymbolProvider
```

The IDynamicSymbolLoader type exposes the following members.

Properties

	Name	Description
	BuildInTypes [▶ 1646]	Gets the build in types. (Inherited from ISymbolLoader [▶ 1645].)
	DataTypes [▶ 1648]	Gets all data types from all Namespaces (Inherited from ISymbolProvider [▶ 1647].)
	RootNamespaceName [▶ 1648]	Gets the name of the root namespace (Inherited from ISymbolProvider [▶ 1647].)
	Settings [▶ 1646]	Gets or sets the access Method (Inherited from ISymbolLoader [▶ 1645].)
	Symbols [▶ 1649]	Gets the (root) symbols of the Symbol provider. (Inherited from ISymbolProvider [▶ 1647].)
	SymbolsDynamic [▶ 1536]	Gets the symbols (late bound as dynamic objects)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.37.1 IDynamicSymbolLoader Properties

The IDynamicSymbolLoader [▶ 1535] type exposes the following members.

Properties

	Name	Description
	BuildInTypes [▶ 1646]	Gets the build in types. (Inherited from ISymbolLoader [▶ 1645].)

	Name	Description
	DataTypes [▶ 1648]	Gets all data types from all Namespaces (Inherited from ISymbolProvider [▶ 1647].)
	RootNamespaceName [▶ 1648]	Gets the name of the root namespace (Inherited from ISymbolProvider [▶ 1647].)
	Settings [▶ 1646]	Gets or sets the access Method (Inherited from ISymbolLoader [▶ 1645].)
	Symbols [▶ 1649]	Gets the (root) symbols of the Symbol provider. (Inherited from ISymbolProvider [▶ 1647].)
	SymbolsDynamic [▶ 1536]	Gets the symbols (late bound as dynamic objects)

Reference

IDynamicSymbolLoader Interface [▶ 1535]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.37.1.1 IDynamicSymbolLoader.SymbolsDynamic Property

Gets the symbols (late bound as dynamic objects)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
DynamicSymbolsContainer SymbolsDynamic { get; }
```

VB

```
ReadOnly Property SymbolsDynamic As DynamicSymbolsContainer
    Get
```

Property Value

Type: DynamicSymbolsContainer [▶ 1388]
The dynamic symbols.

Reference

IDynamicSymbolLoader Interface [▶ 1535]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.38 IEnumType Interface

Common Enum type interface

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IEnumType : IAliasType,
    IDataType, IBitSize
```

VB

```
Public Interface IEnumType
    Inherits IAliasType, IDataType, IBitSize
```

The IEnumType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 1495]	Gets the Base Type (Inherited from IAliasType [▶ 1493].)
	BaseTypeName [▶ 1496]	Gets the BaseType name (Inherited from IAliasType [▶ 1493].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	EnumValues [▶ 1539]	Enumeration specification (if enum)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)

	Name	Description
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Methods

	Name	Description
	Contains [▶ 1540]	Determines whether the enum values contains the specified name
	GetNames [▶ 1541]	Gets the filed names of the IEnumType.T. [▶ 1543]
	GetValues [▶ 1541]	Gets the values of the IEnumType.T. [▶ 1543]
	Parse [▶ 1541]	Parses a value name of the IEnumType.T. [▶ 1543] and returns the value (as base type)
	ToString [▶ 1542]	Returns a String that represents the specified value.
	TryParse [▶ 1543]	Parses the value from value name.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.38.1 IEnumType Properties

The IEnumType [▶ 1536] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 1495]	Gets the Base Type (Inherited from IAliasType [▶ 1493].)
	BaseTypeName [▶ 1496]	Gets the BaseType name (Inherited from IAliasType [▶ 1493].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	EnumValues [▶ 1539]	Enumeration specification (if enum)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)

	Name	Description
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

IEnumType Interface [▶ 1536]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.38.1.1 IEnumType.EnumValues Property

Enumeration specification (if enum)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyEnumValueCollection EnumValues { get; }
```

VB

```
ReadOnly Property EnumValues As ReadOnlyEnumValueCollection
    Get
```

Property Value

Type: ReadOnlyEnumValueCollection [▶ 1736]
The enum specification.

Reference

IEnumType Interface [▶ 1536]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.38.2 IEnumType Methods

The IEnumType [▶ 1536] type exposes the following members.

Methods

	Name	Description
	Contains [▶ 1540]	Determines whether the enum values contains the specified name
	GetNames [▶ 1541]	Gets the filed names of the IEnumType.T. [▶ 1543]
	GetValues [▶ 1541]	Gets the values of the IEnumType.T. [▶ 1543]
	Parse [▶ 1541]	Parses a value name of the IEnumType.T. [▶ 1543] and returns the value (as base type)
	ToString [▶ 1542]	Returns a String that represents the specified value.
	TryParse [▶ 1543]	Parses the value from value name.

Reference

IEnumType Interface [▶ 1536]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.38.2.1 IEnumType.Contains Method

Determines whether the enum values contains the specified name

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool Contains(
    string name
)
```

VB

```
Function Contains (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name.

Return Value

Type: Boolean
true if contains the value, otherwise, false.

Reference

IEnumType Interface [▶ 1536]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.38.2.2 IEnumType.GetNames Method

Gets the filed names of the IEnumType.T. [▶ 1543]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string[] GetNames()
```

VB

```
Function GetNames As String()
```

Return Value

Type: .String.
System.String[].

Reference

IEnumType Interface [▶ 1536]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.38.2.3 IEnumType.GetValues Method

Gets the values of the IEnumType.T. [▶ 1543]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object[] GetValues()
```

VB

```
Function GetValues As Object()
```

Return Value

Type: .Object.
T[].

Reference

IEnumType Interface [▶ 1536]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.38.2.4 IEnumType.Parse Method

Parses a value name of the IEnumType.T. [▶ 1543] and returns the value (as base type)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object Parse(
    string name
)
```

VB

```
Function Parse (
    name As String
) As Object
```

Parameters

name	Type: System.String The value name.

Return Value

Type: Object
T.

Reference

IEnumType Interface [▶ 1536]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.38.2.5 IEnumType.ToString Method

Returns a String that represents the specified value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string ToString(
    Object val
)
```

VB

```
Function ToString (
    val As Object
) As String
```

Parameters

val	Type: System.Object The value.

Return Value

Type: String
A String that represents this value.

Reference

IEnumType Interface [▶ 1536]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.38.2.6 IEnumType.TryParse Method

Parses the value from value name.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TryParse(
    string name,
    out Object value
)
```

VB

```
Function TryParse (
    name As String,
    <OutAttribute> ByRef value As Object
) As Boolean
```

Parameters

name	Type: System.String The value name.
value	Type: System.Object. The value.

Return Value

Type: Boolean
true if value name was found, false otherwise.

Reference

IEnumType Interface [▶ 1536]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.39 IEnumType.T. Interface

Interface representing an enum type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IEnumType<T> : IAliasType,
    IDataType, IBitSize
where T : IConvertible
```

VB

```
Public Interface IEnumType(Of T As IConvertible)
    Inherits IAliasType, IDataType, IBitSize
```

Type Parameters

T	Base type of the Enum

The IEnumType.T. type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 1495]	Gets the Base Type (Inherited from IAliasType [▶ 1493].)
	BaseTypeName [▶ 1496]	Gets the BaseType name (Inherited from IAliasType [▶ 1493].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	EnumValues [▶ 1546]	Enumeration specification (if enum)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)

	Name	Description
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Methods

	Name	Description
	Contains [▶ 1547]	Determines whether the enum values contains the specified name
	GetNames [▶ 1548]	Gets the filed names of the IEnumType.T.
	GetValues [▶ 1548]	Gets the values of the IEnumType.T.
	Parse [▶ 1548]	Parses a name of the IEnumType.T. and returns the value (as base type)
	ToString [▶ 1549]	Returns a String that represents the specified value.
	TryParse [▶ 1550]	Parses the value from value name.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.39.1 IEnumType.T. Properties

The IEnumType.T. [▶ 1543] generic type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 1495]	Gets the Base Type (Inherited from IAliasType [▶ 1493].)
	BaseTypeName [▶ 1496]	Gets the BaseType name (Inherited from IAliasType [▶ 1493].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	EnumValues [▶ 1546]	Enumeration specification (if enum)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)

	Name	Description
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

IEnumType.T. Interface [▶ 1543]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.39.1.1 IEnumType.T..EnumValues Property

Enumeration specification (if enum)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyEnumValueCollection<T> EnumValues { get; }
```

VB

```
ReadOnly Property EnumValues As ReadOnlyEnumValueCollection(Of T)
    Get
```

Property Value

Type: ReadOnlyEnumValueCollection [▶ 1743].T [▶ 1543].
The enum specification.

Reference

IEnumType.T. Interface [▶ 1543]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.39.2 IEnumType.T. Methods

The IEnumType.T. [▶ 1543] generic type exposes the following members.

Methods

	Name	Description
	Contains [▶ 1547]	Determines whether the enum values contains the specified name
	GetNames [▶ 1548]	Gets the filed names of the IEnumType.T. [▶ 1543]
	GetValues [▶ 1548]	Gets the values of the IEnumType.T. [▶ 1543]
	Parse [▶ 1548]	Parses a name of the IEnumType.T. [▶ 1543] and returns the value (as base type)
	ToString [▶ 1549]	Returns a String that represents the specified value.
	TryParse [▶ 1550]	Parses the value from value name.

Reference

IEnumType.T. Interface [▶ 1543]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.39.2.1 IEnumType.T..Contains Method

Determines whether the enum values contains the specified name

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool Contains(
    string name
)
```

VB

```
Function Contains (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name.

Return Value

Type: Boolean
true if contains the value, otherwise, false.

Reference

IEnumType.T. Interface [▶ 1543]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.39.2.2 IEnumType.T..GetNames Method

Gets the filed names of the IEnumType.T. [▶ 1543]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string[] GetNames()
```

VB

```
Function GetNames As String()
```

Return Value

Type: .String.
System.String[].

Reference

IEnumType.T. Interface [▶ 1543]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.39.2.3 IEnumType.T..GetValues Method

Gets the values of the IEnumType.T. [▶ 1543]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
T[] GetValues()
```

VB

```
Function GetValues As T()
```

Return Value

Type: .T [▶ 1543].
T[].

Reference

IEnumType.T. Interface [▶ 1543]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.39.2.4 IEnumType.T..Parse Method

Parses a name of the IEnumType.T. [▶ 1543] and returns the value (as base type)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
T Parse(
    string name
)
```

VB

```
Function Parse (
    name As String
) As T
```

Parameters

name	Type: System.String The name.

Return Value

Type: T [▶ 1543]
T.

Reference

IEnumType.T. Interface [▶ 1543]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.39.2.5 IEnumType.T..ToString Method

Returns a String that represents the specified value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string ToString(
    T val
)
```

VB

```
Function ToString (
    val As T
) As String
```

Parameters

val	Type: T [▶ 1543] The value.

Return Value

Type: String
A String that represents this value.

Reference

IEnumType.T. Interface [▶ 1543]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.39.2.6 IEnumType.T..TryParse Method

Parses the value from value name.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TryParse(
    string name,
    out T value
)
```

VB

```
Function TryParse (
    name As String,
    <OutAttribute> ByRef value As T
) As Boolean
```

Parameters

name	Type: System.String The value name.
value	Type: T [▶ 1543]. The value.

Return Value

Type: Boolean
true if value name was found, false otherwise.

Reference

IEnumType.T. Interface [▶ 1543]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.40 IEnumValue Interface

Generic interface for EnumValues

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IEnumValue
```

VB

```
Public Interface IEnumValue
```

The IEnumValue type exposes the following members.

Properties

	Name	Description
	ManagedBaseType [▶ 1551]	Gets the enumeration base type (sint,byte,short,ushort,int,uint,Int64,UInt64 supported)
	Name [▶ 1552]	Gets the name of the Enumeration Value (value as string)
	Primitive [▶ 1552]	Gets the (Primitive, BaseType) Value of the enumeration as object
	RawValue [▶ 1553]	Gets the raw value of the enumeration (as byte array)
	Size [▶ 1553]	Gets the size of the Enum value (in bytes)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.40.1 IEnumValue Properties

The IEnumValue [▶ 1550] type exposes the following members.

Properties

	Name	Description
	ManagedBaseType [▶ 1551]	Gets the enumeration base type (sint,byte,short,ushort,int,uint,Int64,UInt64 supported)
	Name [▶ 1552]	Gets the name of the Enumeration Value (value as string)
	Primitive [▶ 1552]	Gets the (Primitive, BaseType) Value of the enumeration as object
	RawValue [▶ 1553]	Gets the raw value of the enumeration (as byte array)
	Size [▶ 1553]	Gets the size of the Enum value (in bytes)

Reference

IEnumValue Interface [▶ 1550]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.40.1.1 IEnumValue.ManagedBaseType Property

Gets the enumeration base type (sint,byte,short,ushort,int,uint,Int64,UInt64 supported)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Type ManagedBaseType { get; }
```

VB

```
ReadOnly Property ManagedBaseType As Type
    Get
```

Property Value

Type: Type
The type of the base.

Reference

IEnumValue Interface [▶ 1550]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.40.1.2 IEnumValue.Name Property

Gets the name of the Enumeration Value (value as string)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Name { get; }
```

VB

```
ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Reference

IEnumValue Interface [▶ 1550]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.40.1.3 IEnumValue.Primitive Property

Gets the (Primitive, BaseType) Value of the enumeration as object

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object Primitive { get; }
```

VB

```
ReadOnly Property Primitive As Object
    Get
```

Property Value

Type: Object
The object value.

Reference

IEnumValue Interface [▶ 1550]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.40.1.4 IEnumValue.RawValue Property

Gets the raw value of the enumeration (as byte array)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
byte[] RawValue { get; }
```

VB

```
ReadOnly Property RawValue As Byte()
    Get
```

Property Value

Type: .Byte.
The raw value.

Reference

IEnumValue Interface [▶ 1550]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.40.1.5 IEnumValue.Size Property

Gets the size of the Enum value (in bytes)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Size { get; }
```

VB

```
ReadOnly Property Size As Integer
    Get
```

Property Value

Type: Int32
The size.

Reference

IEnumValue Interface [▶ 1550]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.41 IField Interface

Specifies a single field/member of a Struct DataType [▶ 1619].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IField : IAttributedInstance, IInstance,
    IBitSize
```

VB

```
Public Interface IField
    Inherits IAttributedInstance, IInstance, IBitSize
```

The IField type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)

	Name	Description
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	ParentType [▶ 1556]	Gets the Parent Struct/Union of this IField.
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.41.1 IField Properties

The IField [▶ 1554] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)

	Name	Description
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	ParentType [▶ 1556]	Gets the Parent Struct/Union of this IField [▶ 1554].
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IField Interface [▶ 1554]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.41.1.1 IField.ParentType Property

Gets the Parent Struct/Union of this IField [▶ 1554].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType ParentType { get; }
```

VB

```
ReadOnly Property ParentType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the parent.

Reference

IField Interface [▶ 1554]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.42 IInstance Interface

Interface specifying instance objects.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IInstance : IBitSize
```

VB

```
Public Interface IInstance
    Inherits IBitSize
```

The IInstance type exposes the following members.

Properties

	Name	Description
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Comment [▶ 1558]	Gets the comment of the IInstance
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance.
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.))
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsPointer [▶ 1560]	Indicates that the IInstance represents a Pointer type (Pointer TO)
	IsReference [▶ 1561]	Indicates that the IInstance represents a Reference type (REFERENCE TO)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance is static.
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.42.1 IInstance Properties

The IInstance [▶ 1556] type exposes the following members.

Properties

	Name	Description
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)

	Name	Description
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556]
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556].
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.))
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static.
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556].

Reference

IInstance Interface [▶ 1556]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.42.1.1 IInstance.Comment Property

Gets the comment of the IInstance [▶ 1556]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Comment { get; }
```

VB

```
ReadOnly Property Comment As String
    Get
```

Property Value

Type: String
The comment.

Reference

IInstance Interface [▶ 1556]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.42.1.2 IInstance.DataType Property

Gets the IDataType [▶ 1517] of the IInstance [▶ 1556].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType DataType { get; }
```

VB

```
ReadOnly Property DataType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the data.

Reference

IInstance Interface [▶ 1556]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.42.1.3 IInstance.InstanceName Property

Gets the name of the instance (without periods (.)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string InstanceName { get; }
```

VB

```
ReadOnly Property InstanceName As String
    Get
```

Property Value

Type: String
The name of the instance.

Reference

IInstance Interface [▶ 1556]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.42.1.4 IInstance.InstancePath Property

Gets the relative / absolute access path to the instance (with periods (.))

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string InstancePath { get; }
```

VB

```
ReadOnly Property InstancePath As String
    Get
```

Property Value

Type: String
The instance path.

Remarks

If this path is relative or absolute depends on the context. IMember [▶ 1562] are using relative paths, ISymbol [▶ 1634]s are using absolute ones.

Reference

IInstance Interface [▶ 1556]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.42.1.5 IInstance.IsPointer Property

Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsPointer { get; }
```

VB

```
ReadOnly Property IsPointer As Boolean
    Get
```

Property Value

Type: Boolean
true if is ReferenceTo, otherwise false.

Reference

IInstance Interface [▶ 1556]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.42.1.6 IInstance.IsReference Property

Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsReference { get; }
```

VB

```
ReadOnly Property IsReference As Boolean
    Get
```

Property Value

Type: Boolean
true if is ReferenceTo, otherwise false.

Reference

IInstance Interface [▶ 1556]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.42.1.7 IInstance.IsStatic Property

Gets a value indicating whether this IInstance [▶ 1556] is static.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsStatic { get; }
```

VB

```
ReadOnly Property IsStatic As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is static; otherwise, false.

Reference

IInstance Interface [▶ 1556]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.42.1.8 IInstance.TypeName Property

Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string TypeName { get; }
```

VB

```
ReadOnly Property TypeName As String
    Get
```

Property Value

Type: String
The name of the type.

Reference

IInstance Interface [▶ 1556]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.43 IMember Interface

Specifies a single field/member of a Struct DataType [▶ 1619].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IMember : IField,
    IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IMember
    Inherits IField, IAttributedInstance, IInstance, IBitSize
```

The IMember type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitOffset [▶ 1564]	Gets the bit offset.

	Name	Description
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteOffset [▶ 1565]	Gets the byte offset.
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Offset [▶ 1565]	Gets the offset of the IMember within the parent IStructType [▶ 1619] in bits or bytes dependent on IsBitType.
	ParentType [▶ 1556]	Gets the Parent Struct/Union of this IField [▶ 1554]. (Inherited from IField [▶ 1554].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.43.1 IMember Properties

The IMember [▶ 1562] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitOffset [▶ 1564]	Gets the bit offset.

	Name	Description
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteOffset [▶ 1565]	Gets the byte offset.
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Offset [▶ 1565]	Gets the offset of the IMember [▶ 1562] within the parent IStructType [▶ 1619] in bits or bytes dependent on IsBitType.
	ParentType [▶ 1556]	Gets the Parent Struct/Union of this IField [▶ 1554]. (Inherited from IField [▶ 1554].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IMember Interface [▶ 1562]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.43.1.1 IMember.BitOffset Property

Gets the bit offset.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int BitOffset { get; }
```

VB

```
ReadOnly Property BitOffset As Integer
    Get
```

Property Value

Type: Int32
The bit offset.

Reference

IMember Interface [▶ 1562]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.43.1.2 IMember.ByteOffset Property

Gets the byte offset.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int ByteOffset { get; }
```

VB

```
ReadOnly Property ByteOffset As Integer
    Get
```

Property Value

Type: Int32
The byte offset.

Reference

IMember Interface [▶ 1562]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.43.1.3 IMember.Offset Property

Gets the offset of the IMember [▶ 1562] within the parent IStructType [▶ 1619] in bits or bytes dependent on IsBitType.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Offset { get; }
```

VB

```
ReadOnly Property Offset As Integer
    Get
```

Property Value

Type: Int32
The bit offset.

Reference

IMember Interface [▶ 1562]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.44 INotificationSettings Interface

Interface for Notification Settings

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface INotificationSettings : IComparable<INotificationSettings>
```

VB

```
Public Interface INotificationSettings
    Inherits IComparable(Of INotificationSettings)
```

The INotificationSettings type exposes the following members.

Methods

	Name	Description
	CompareTo	Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object. (Inherited from IComparable.INotificationSettings..)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.44.1 INotificationSettings Methods

The INotificationSettings [▶ 1566] type exposes the following members.

Methods

	Name	Description
	CompareTo	Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object. (Inherited from IComparable.INotificationSettings [▶ 1566]..)

Reference

INotificationSettings Interface [▶ 1566]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.45 IOversamplingArrayInstance Interface

Interface IOversamplingArrayInstance

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IOversamplingArrayInstance : IArrayInstance,
    ISymbol, IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IOversamplingArrayInstance
    Inherits IArrayInstance, ISymbol, IAttributedInstance, IInstance,
    IBitSize
```

The IOversamplingArrayInstance type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	Dimensions [▶ 1499]	Gets the dimensions as read only collection. (Inherited from IArrayInstance [▶ 1496].)
	Elements [▶ 1500]	Gets the contained Array Elements as read only collection. (Inherited from IArrayInstance [▶ 1496].)
	ElementType [▶ 1500]	Gets the type of the contained elements. (Inherited from IArrayInstance [▶ 1496].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)

	Name	Description
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Item [▶ 1501]	Gets the ISymbol [▶ 1634] with the specified indices. (Inherited from IArrayInstance [▶ 1496].)
	OversamplingElement [▶ 1570]	Gets the oversampling element.
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Methods

	Name	Description
	TryGetElement(IList.. Int32.., ISymbol.) [▶ 1502]	Tries to get the array element with the specified indices (jagged array support). (Inherited from IArrayInstance [▶ 1496].)
	TryGetElement(.Int32., ISymbol.) [▶ 1503]	Tries to get the array element with specified indices (only first level on jagged arrays) (Inherited from IArrayInstance [▶ 1496].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IArrayInstance [▶ 1496]

5.8.45.1 IOversamplingArrayInstance Properties

The IOversamplingArrayInstance [▶ 1567] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	Dimensions [▶ 1499]	Gets the dimensions as read only collection. (Inherited from IArrayInstance [▶ 1496].)
	Elements [▶ 1500]	Gets the contained Array Elements as read only collection. (Inherited from IArrayInstance [▶ 1496].)
	ElementType [▶ 1500]	Gets the type of the contained elements. (Inherited from IArrayInstance [▶ 1496].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)

	Name	Description
	Item [▶ 1501]	Gets the ISymbol [▶ 1634] with the specified indices. (Inherited from IArrayInstance [▶ 1496].)
	OversamplingElement [▶ 1570]	Gets the oversampling element.
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IOversamplingArrayInstance Interface [▶ 1567]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.45.1.1 IOversamplingArrayInstance.OversamplingElement Property

Gets the oversampling element.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ISymbol OversamplingElement { get; }
```

VB

```
ReadOnly Property OversamplingElement As ISymbol
    Get
```

Property Value

Type: ISymbol [▶ 1634]
The oversampling element.

Reference

IOversamplingArrayInstance Interface [▶ 1567]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.45.2 IOversamplingArrayInstance Methods

Methods

	Name	Description
	TryGetElement(IList.., Int32.., ISymbol.) [▶ 1502]	Tries to get the array element with the specified indices (jagged array support). (Inherited from IArrayInstance [▶ 1496].)
	TryGetElement(.Int32., ISymbol.) [▶ 1503]	Tries to get the array element with specified indices (only first level on jagged arrays) (Inherited from IArrayInstance [▶ 1496].)

Reference

IOversamplingArrayInstance Interface [▶ 1567]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.46 IPointerInstance Interface

Interface representing an instance of an IPointerType [▶ 1574]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IPointerInstance : ISymbol,
    IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IPointerInstance
    Inherits ISymbol, IAttributedInstance, IInstance, IBitSize
```

The IPointerInstance type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

	Name	Description
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Reference [▶ 1574]	Gets the resolved reference of Pointer / Reference
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.46.1 IPointerInstance Properties

The IPointerInstance [▶ 1571] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)

	Name	Description
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Reference [▶ 1574]	Gets the resolved reference of Pointer / Reference
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IPointerInstance Interface [▶ 1571]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.46.1.1 IPointerInstance.Reference Property

Gets the resolved reference of Pointer / Reference

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ISymbol Reference { get; }
```

VB

```
ReadOnly Property Reference As ISymbol
    Get
```

Property Value

Type: ISymbol [▶ 1634]
The reference.

Reference

IPointerInstance Interface [▶ 1571]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.47 IPointerType Interface

Interface representing a pointer type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IPointerType : IDataType,
    IBitSize
```

VB

```
Public Interface IPointerType
    Inherits IDataType, IBitSize
```

The IPointerType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)

	Name	Description
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	ReferencedType [▶ 1576]	Gets the the referenced type.
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.47.1 IPointerType Properties

The IPointerType [▶ 1574] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)

	Name	Description
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	ReferencedType [▶ 1576]	Gets the the referenced type.
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

IPointerType Interface [▶ 1574]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.47.1.1 IPointerType.ReferencedType Property

Gets the the referenced type.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType ReferencedType { get; }
```

VB

```
ReadOnly Property ReferencedType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the referenced.

Reference

IPointerType Interface [▶ 1574]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.48 IPrimitiveType Interface

Interface IPrimitiveType

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IPrimitiveType : IDataType,
    IBitSize
```

VB

```
Public Interface IPrimitiveType
    Inherits IDataType, IBitSize
```

The IPrimitiveType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)

	Name	Description
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	PrimitiveFlags [▶ 1579]	Indicates types of different PrimitiveTypes with flags.
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IDataType [▶ 1517]

5.8.48.1 IPrimitiveType Properties

The IPrimitiveType [▶ 1577] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)

	Name	Description
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	PrimitiveFlags [▶ 1579]	Indicates types of different PrimitiveTypes with flags.
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

IPrimitiveType Interface [▶ 1577]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.48.1.1 IPrimitiveType.PrimitiveFlags Property

Indicates types of different PrimitiveTypes with flags.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
PrimitiveTypeFlags PrimitiveFlags { get; }
```

VB

```
ReadOnly Property PrimitiveFlags As PrimitiveTypeFlags
    Get
```

Property Value

Type: PrimitiveTypeFlags [▶ 1724]

Reference

IPrimitiveType Interface [▶ 1577]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.49 IProcessImageAddress Interface

Interface describing a Process Image Address

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IProcessImageAddress : IBitSize
```

VB

```
Public Interface IProcessImageAddress
    Inherits IBitSize
```

The IProcessImageAddress type exposes the following members.

Properties

	Name	Description
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	IndexGroup [▶ 1581]	Gets the index group of the Symbol
	IndexOffset [▶ 1581]	Gets the index offset of the Symbol
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.49.1 IProcessImageAddress Properties

The IProcessImageAddress [▶ 1579] type exposes the following members.

Properties

	Name	Description
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	IndexGroup [▶ 1581]	Gets the index group of the Symbol
	IndexOffset [▶ 1581]	Gets the index offset of the Symbol
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)

	Name	Description
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

IProcessImageAddress Interface [▶ 1579]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.49.1.1 IProcessImageAddress.IndexGroup Property

Gets the index group of the Symbol

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
uint IndexGroup { get; }
```

VB

```
ReadOnly Property IndexGroup As UInteger
    Get
```

Property Value

Type: UInt32
The index group.

Reference

IProcessImageAddress Interface [▶ 1579]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.49.1.2 IProcessImageAddress.IndexOffset Property

Gets the index offset of the Symbol

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
uint IndexOffset { get; }
```

VB

```
ReadOnly Property IndexOffset As UInteger
    Get
```

Property Value

Type: UInt32
The index offset.

Reference

IProcessImageAddress Interface [▶ 1579]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.50 IReferenceInstance Interface

Interface representing an instance of an IReferenceType [▶ 1587]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IReferenceInstance : ISymbol,
    IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IReferenceInstance
    Inherits ISymbol, IAttributedInstance, IInstance, IBitSize
```

The IReferenceInstance type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)

	Name	Description
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	ReferencedType [▶ 1585]	Gets the referenced type
	ResolvedByteSize [▶ 1585]	Get the ByteSize of the (completely) resolved Symbol
	ResolvedCategory [▶ 1586]	Gets the Category of the Referenced Symbol.
	ResolvedType [▶ 1586]	Gets the (completely) resolved type
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.50.1 IReferenceInstance Properties

The IReferenceInstance [▶ 1582] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)

	Name	Description
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	ReferencedType [▶ 1585]	Gets the referenced type
	ResolvedByteSize [▶ 1585]	Get the ByteSize of the (completely) resolved Symbol
	ResolvedCategory [▶ 1586]	Gets the Category of the Referenced Symbol.
	ResolvedType [▶ 1586]	Gets the (completely) resolved type
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IReferenceInstance Interface [▶ 1582]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.50.1.1 IReferenceInstance.ReferencedType Property

Gets the referenced type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType ReferencedType { get; }
```

VB

```
ReadOnly Property ReferencedType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the referenced type

Remarks

This is no complete resolvement, only the next level. The referenced type can be a reference again.

Reference

IReferenceInstance Interface [▶ 1582]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.50.1.2 IReferenceInstance.ResolvedByteSize Property

Get the ByteSize of the (completely) resolved Symbol

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int ResolvedByteSize { get; }
```

VB

```
ReadOnly Property ResolvedByteSize As Integer
    Get
```

Property Value

Type: Int32
The size of the resolved byte.

Reference

IReferenceInstance Interface [▶ 1582]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.50.1.3 IReferenceInstance.ResolvedCategory Property

Gets the Category of the Referenced Symbol.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
DataTypeCategory ResolvedCategory { get; }
```

VB

```
ReadOnly Property ResolvedCategory As DataTypeCategory
    Get
```

Property Value

Type: DataTypeCategory [▶ 1189]
The resolved category.

Reference

IReferenceInstance Interface [▶ 1582]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.50.1.4 IReferenceInstance.ResolvedType Property

Gets the (completely) resolved type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType ResolvedType { get; }
```

VB

```
ReadOnly Property ResolvedType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the resolved symbol

Reference

IReferenceInstance Interface [▶ 1582]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.51 IReferenceType Interface

Interface representing a reference/pointer type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IReferenceType : IDataType,
    IBitSize
```

VB

```
Public Interface IReferenceType
    Inherits IDataType, IBitSize
```

The IReferenceType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)

	Name	Description
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	ReferencedType [▶ 1589]	Gets the the referenced type.
	ResolvedByteSize [▶ 1589]	Get the ByteSize of the (completely) resolved Symbol
	ResolvedCategory [▶ 1590]	Gets the Category of the (completely) resolved Symbol.
	ResolvedType [▶ 1590]	Gets the (completely) resolved type
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.51.1 IReferenceType Properties

The IReferenceType [▶ 1587] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)

	Name	Description
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	ReferencedType [▶ 1589]	Gets the the referenced type.
	ResolvedByteSize [▶ 1589]	Get the ByteSize of the (completely) resolved Symbol
	ResolvedCategory [▶ 1590]	Gets the Category of the (completely) resolved Symbol.
	ResolvedType [▶ 1590]	Gets the (completely) resolved type
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

IReferenceType Interface [▶ 1587]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.51.1.1 IReferenceType.ReferencedType Property

Gets the the referenced type.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType ReferencedType { get; }
```

VB

```
ReadOnly Property ReferencedType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the referenced.

Reference

IReferenceType Interface [▶ 1587]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.51.1.2 IReferenceType.ResolvedByteSize Property

Get the ByteSize of the (completely) resolved Symbol

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int ResolvedByteSize { get; }
```

VB

```
ReadOnly Property ResolvedByteSize As Integer
    Get
```

Property Value

Type: Int32
The size of the resolved byte.

Reference

IReferenceType Interface [▶ 1587]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.51.1.3 IReferenceType.ResolvedCategory Property

Gets the Category of the (completely) resolved Symbol.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
DataTypeCategory ResolvedCategory { get; }
```

VB

```
ReadOnly Property ResolvedCategory As DataTypeCategory
    Get
```

Property Value

Type: DataTypeCategory [▶ 1189]
The resolved category.

Reference

IReferenceType Interface [▶ 1587]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.51.1.4 IReferenceType.ResolvedType Property

Gets the (completely) resolved type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType ResolvedType { get; }
```

VB

```
ReadOnly Property ResolvedType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the resolved symbol

Reference

IReferenceType Interface [▶ 1587]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.52 IRpcCallableInstance Interface

Interface for an RPC callable PLC Method (Remote procedure call)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IRpcCallableInstance
```

VB

```
Public Interface IRpcCallableInstance
```

The IRpcCallableInstance type exposes the following members.

Properties

	Name	Description
	RpcMethods [▶ 1592]	Gets the Method descriptions for the IRpcCallableType [▶ 1596]

Methods

	Name	Description
	InvokeRpcMethod [▶ 1593]	Invokes the specified method.
	TryInvokeRpcMethod(String, .Object., Object.) [▶ 1594]	Tries to invoke the specified method.

	Name	Description
	TryInvokeRpcMethod(IRpcMethod, .Object., Object.) [▶ 1595]	Tries to invoke the specified method.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.52.1 IRpcCallableInstance Properties

The IRpcCallableInstance [▶ 1591] type exposes the following members.

Properties

	Name	Description
	RpcMethods [▶ 1592]	Gets the Method descriptions for the IRpcCallableType [▶ 1596]

Reference

IRpcCallableInstance Interface [▶ 1591]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.52.1.1 IRpcCallableInstance.RpcMethods Property

Gets the Method descriptions for the IRpcCallableType [▶ 1596]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyRpcMethodCollection RpcMethods { get; }
```

VB

```
ReadOnly Property RpcMethods As ReadOnlyRpcMethodCollection
    Get
```

Property Value

Type: ReadOnlyRpcMethodCollection [▶ 1762]
The methods.

Reference

IRpcCallableInstance Interface [▶ 1591]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.52.2 IRpcCallableInstance Methods

The IRpcCallableInstance [▶ 1591] type exposes the following members.

Methods

	Name	Description
	InvokeRpcMethod [▶ 1593]	Invokes the specified method.
	TryInvokeRpcMethod(String, .Object., Object.) [▶ 1594]	Tries to invoke the specified method.
	TryInvokeRpcMethod(IRpcMethod, .Object., Object.) [▶ 1595]	Tries to invoke the specified method.

Reference

IRpcCallableInstance Interface [▶ 1591]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.52.2.1 IRpcCallableInstance.InvokeRpcMethod Method

Invokes the specified method.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object InvokeRpcMethod(
    string methodName,
    Object[] parameters
)
```

VB

```
Function InvokeRpcMethod (
    methodName As String,
    parameters As Object()
) As Object
```

Parameters

methodName	Type: System.String Name of the method.
parameters	Type: .System.Object. The parameters.

Return Value

Type: Object
The return value of the RPC Method

Remarks

To indicate a PLC Method for remote ads access, the attribute 'TcRpcEnable' must be declared on the method declaration (see example).

Examples

RPC Method definition and implementation

```
(* Declaration *)
{attribute 'TcRpcEnable'}
METHOD RpcMethod1 : INT
VAR_INPUT
   i1 : INT;
END_VAR

(* Implementation *)
RpcMethod1 := i1 + 1;
```

Reference

IRpcCallableInstance Interface [▶ 1591]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.52.2.2 IRpcCallableInstance.TryInvokeRpcMethod Method

Overload List

	Name	Description
	TryInvokeRpcMethod(String, .Object., Object.) [▶ 1594]	Tries to invoke the specified method.
	TryInvokeRpcMethod(IRpcMethod, .Object., Object.) [▶ 1595]	Tries to invoke the specified method.

Reference

IRpcCallableInstance Interface [▶ 1591]

TwinCAT.TypeSystem Namespace [▶ 1184]

IRpcCallableInstance.TryInvokeRpcMethod Method (String, .Object., Object.)

Tries to invoke the specified method.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int TryInvokeRpcMethod(
    string methodName,
    Object[] args,
    out Object result
)
```

VB

```
Function TryInvokeRpcMethod (
    methodName As String,
    args As Object(),
    <OutAttribute> ByRef result As Object
) As Integer
```

Parameters

methodName	Type: System.String Name of the method.
args	Type: .System.Object. The arguments.
result	Type: System.Object. The result.

Return Value

Type: Int32
true if the call succeeds, false otherwise.

Reference

IRpcCallableInstance Interface [▶ 1591]

TryInvokeRpcMethod Overload [▶ 1594]

TwinCAT.TypeSystem Namespace [▶ 1184]

IRpcCallableInstance.TryInvokeRpcMethod Method (IRpcMethod, .Object., Object.)

Tries to invoke the specified method.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int TryInvokeRpcMethod(
    IRpcMethod method,
    Object[] args,
    out Object result
)
```

VB

```
Function TryInvokeRpcMethod (
    method As IRpcMethod,
    args As Object(),
    <OutAttribute> ByRef result As Object
) As Integer
```

Parameters

method	Type: TwinCAT.TypeSystem.IRpcMethod [▶ 1597] The method.
args	Type: .System.Object. The arguments.
result	Type: System.Object. The result.

Return Value

Type: Int32
true if the call succeeds, false otherwise.

Reference

IRpcCallableInstance Interface [▶ 1591]

TryInvokeRpcMethod Overload [▶ 1594]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.53 IRpcCallableType Interface

Interface representing an RPC callable IStructType [▶ 1619]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IRpcCallableType
```

VB

```
Public Interface IRpcCallableType
```

The IRpcCallableType type exposes the following members.

Properties

	Name	Description
	RpcMethods [▶ 1597]	Gets the Method descriptions for the IRpcCallableType

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.53.1 IRpcCallableType Properties

The IRpcCallableType [▶ 1596] type exposes the following members.

Properties

	Name	Description
	RpcMethods [▶ 1597]	Gets the Method descriptions for the IRpcCallableType [▶ 1596]

Reference

IRpcCallableType Interface [▶ 1596]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.53.1.1 IRpcCallableType.RpcMethods Property

Gets the Method descriptions for the IRpcCallableType [▶ 1596]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyRpcMethodCollection RpcMethods { get; }
```

VB

```
ReadOnly Property RpcMethods As ReadOnlyRpcMethodCollection
    Get
```

Property Value

Type: ReadOnlyRpcMethodCollection [▶ 1762]
The RPC methods.

Remarks

The DataType (Structure) must be marked with the PlcAttribute 'TcRpcEnable' to enable RpcMethods, otherwise RpcMethods are not passed through to the ADS symbolic information.

Reference

IRpcCallableType Interface [▶ 1596]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.54 IRpcMethod Interface

Interface describes an RPC Method

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IRpcMethod
```

VB

```
Public Interface IRpcMethod
```

The IRpcMethod type exposes the following members.

Properties

	Name	Description
	Comment [▶ 1598]	Gets the Method comment.
	IsVoid [▶ 1599]	Gets a value indicating whether this IRpcMethod has no return parameter

	Name	Description
	Name [▶ 1599]	Gets the name of the method
	Parameters [▶ 1600]	Gets the Method parameter descriptions.
	ReturnType [▶ 1600]	Gets the return type.
	ReturnTypeSize [▶ 1601]	Gets the size of the return type in bytes.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.54.1 IRpcMethod Properties

The IRpcMethod [▶ 1597] type exposes the following members.

Properties

	Name	Description
	Comment [▶ 1598]	Gets the Method comment.
	IsVoid [▶ 1599]	Gets a value indicating whether this IRpcMethod [▶ 1597] has no return parameter
	Name [▶ 1599]	Gets the name of the method
	Parameters [▶ 1600]	Gets the Method parameter descriptions.
	ReturnType [▶ 1600]	Gets the return type.
	ReturnTypeSize [▶ 1601]	Gets the size of the return type in bytes.

Reference

IRpcMethod Interface [▶ 1597]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.54.1.1 IRpcMethod.Comment Property

Gets the Method comment.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Comment { get; }
```

VB

```
ReadOnly Property Comment As String
    Get
```

Property Value

Type: String
The comment.

Reference

IRpcMethod Interface [▶ 1597]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.54.1.2 IRpcMethod.IsVoid Property

Gets a value indicating whether this IRpcMethod [▶ 1597] has no return parameter

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsVoid { get; }
```

VB

```
ReadOnly Property IsVoid As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is void; otherwise, false.

Reference

IRpcMethod Interface [▶ 1597]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.54.1.3 IRpcMethod.Name Property

Gets the name of the method

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Name { get; }
```

VB

```
ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Reference

IRpcMethod Interface [▶ 1597]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.54.1.4 IRpcMethod.Parameters Property

Gets the Method parameter descriptions.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyMethodParameterCollection Parameters { get; }
```

VB

```
ReadOnly Property Parameters As ReadOnlyMethodParameterCollection
    Get
```

Property Value

Type: ReadOnlyMethodParameterCollection [▶ 1759]
The parameters.

Reference

IRpcMethod Interface [▶ 1597]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.54.1.5 IRpcMethod.ReturnType Property

Gets the return type.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string ReturnType { get; }
```

VB

```
ReadOnly Property ReturnType As String
    Get
```

Property Value

Type: String
Return type.

Reference

IRpcMethod Interface [▶ 1597]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.54.1.6 IRpcMethod.ReturnTypeSize Property

Gets the size of the return type in bytes.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int ReturnTypeSize { get; }
```

VB

```
ReadOnly Property ReturnTypeSize As Integer
    Get
```

Property Value

Type: Int32
The size of the return type.

Reference

IRpcMethod Interface [▶ 1597]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.55 IRpcMethodParameter Interface

Interface IRpcMethodParameter

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IRpcMethodParameter
```

VB

```
Public Interface IRpcMethodParameter
```

The IRpcMethodParameter type exposes the following members.

Properties

	Name	Description
	Name [▶ 1602]	Gets the Parameter Name
	ParameterFlags [▶ 1603]	Gets the parameter flags.
	Size [▶ 1603]	Gets the size of the IRpcMethodParameter
	TypeName [▶ 1604]	Gets the Data type of the Parameter

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.55.1 IRpcMethodParameter Properties

The IRpcMethodParameter [▶ 1601] type exposes the following members.

Properties

	Name	Description
	Name [▶ 1602]	Gets the Parameter Name
	ParameterFlags [▶ 1603]	Gets the parameter flags.
	Size [▶ 1603]	Gets the size of the IRpcMethodParameter [▶ 1601]
	TypeName [▶ 1604]	Gets the Data type of the Parameter

Reference

IRpcMethodParameter Interface [▶ 1601]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.55.1.1 IRpcMethodParameter.Name Property

Gets the Parameter Name

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Name { get; }
```

VB

```
ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Reference

IRpcMethodParameter Interface [▶ 1601]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.55.1.2 IRpcMethodParameter.ParameterFlags Property

Gets the parameter flags.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
MethodParamFlags ParameterFlags { get; }
```

VB

```
ReadOnly Property ParameterFlags As MethodParamFlags
    Get
```

Property Value

Type: MethodParamFlags [▶ 1724]
The parameter flags.

Reference

IRpcMethodParameter Interface [▶ 1601]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.55.1.3 IRpcMethodParameter.Size Property

Gets the size of the IRpcMethodParameter [▶ 1601]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Size { get; }
```

VB

```
ReadOnly Property Size As Integer
    Get
```

Property Value

Type: Int32
The size.

Reference

IRpcMethodParameter Interface [▶ 1601]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.55.1.4 IRpcMethodParameter.TypeName Property

Gets the Data type of the Parameter

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string TypeName { get; }
```

VB

```
ReadOnly Property TypeName As String
    Get
```

Property Value

Type: String
The type.

Reference

IRpcMethodParameter Interface [▶ 1601]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.56 IRpcStructInstance Interface

Interface IRpcStructInstance

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IRpcStructInstance : IStructInstance,
    ISymbol, IAttributedInstance, IInstance, IBitSize, IRpcCallableInstance
```

VB

```
Public Interface IRpcStructInstance
    Inherits IStructInstance, ISymbol, IAttributedInstance, IInstance,
    IBitSize, IRpcCallableInstance
```

The IRpcStructInstance type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasRpcMethods [▶ 1618]	Gets a value indicating whether this instance has RPC methods (Inherited from IStructInstance [▶ 1615].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	MemberInstances [▶ 1619]	Gets the member instances of the Struct Instance [▶ 1615]. (Inherited from IStructInstance [▶ 1615].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	RpcMethods [▶ 1592]	Gets the Method descriptions for the IRpcCallableType [▶ 1596] (Inherited from IRpcCallableInstance [▶ 1591].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

	Name	Description
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Methods

	Name	Description
	InvokeRpcMethod [▶ 1593]	Invokes the specified method. (Inherited from IRpcCallableInstance [▶ 1591].)
	TryInvokeRpcMethod(String, .Object., Object.) [▶ 1594]	Tries to invoke the specified method. (Inherited from IRpcCallableInstance [▶ 1591].)
	TryInvokeRpcMethod(IRpcMethod, .Object., Object.) [▶ 1595]	Tries to invoke the specified method. (Inherited from IRpcCallableInstance [▶ 1591].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IStructInstance [▶ 1615]

TwinCAT.TypeSystem.IRpcCallableInstance [▶ 1591]

5.8.56.1 IRpcStructInstance Properties

The IRpcStructInstance [▶ 1604] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasRpcMethods [▶ 1618]	Gets a value indicating whether this instance has RPC methods (Inherited from IStructInstance [▶ 1615].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)

	Name	Description
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	MemberInstances [▶ 1619]	Gets the member instances of the Struct Instance [▶ 1615]. (Inherited from IStructInstance [▶ 1615].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	RpcMethods [▶ 1592]	Gets the Method descriptions for the IRpcCallableType [▶ 1596] (Inherited from IRpcCallableInstance [▶ 1591].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IRpcStructInstance Interface [▶ 1604]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.56.2 IRpcStructInstance Methods

The IRpcStructInstance [▶ 1604] type exposes the following members.

Methods

	Name	Description
	InvokeRpcMethod [▶ 1593]	Invokes the specified method. (Inherited from IRpcCallableInstance [▶ 1591].)
	TryInvokeRpcMethod(String, .Object., Object.) [▶ 1594]	Tries to invoke the specified method. (Inherited from IRpcCallableInstance [▶ 1591].)
	TryInvokeRpcMethod(IRpcMethod, .Object., Object.) [▶ 1595]	Tries to invoke the specified method. (Inherited from IRpcCallableInstance [▶ 1591].)

Reference

IRpcStructInstance Interface [▶ 1604]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.57 IStringInstance Interface

Interface IStringInstance

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IStringInstance : ISymbol,
    IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IStringInstance
    Inherits ISymbol, IAttributedInstance, IInstance, IBitSize
```

The IStringInstance type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)

	Name	Description
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	Encoding [▶ 1611]	Gets the encoding of the String (Encoding.Default (STRING) or Encoding.UNICODE (WSTRING))
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsFixedLength [▶ 1611]	Gets a value indicating whether this instance is a string of static length
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.ISymbol [▶ 1634]

TwinCAT.TypeSystem.IStringType [▶ 1612]

5.8.57.1 IStringInstance Properties

The IStringInstance [▶ 1608] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	Encoding [▶ 1611]	Gets the encoding of the String (Encoding.Default (STRING) or Encoding.UNICODE (WSTRING))
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsFixedLength [▶ 1611]	Gets a value indicating whether this instance is a string of static length
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)

	Name	Description
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IStringInstance Interface [▶ 1608]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.57.1.1 IStringInstance.Encoding Property

Gets the encoding of the String (Encoding.Default (STRING) or Encoding.UNICODE (WSTRING))

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Encoding Encoding { get; }
```

VB

```
ReadOnly Property Encoding As Encoding
    Get
```

Property Value

Type: Encoding
The encoding.

Reference

IStringInstance Interface [▶ 1608]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.57.1.2 IStringInstance.IsFixedLength Property

Gets a value indicating whether this instance is a string of static length

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsFixedLength { get; }
```

VB

```
ReadOnly Property IsFixedLength As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is of static length; otherwise, false.

Reference

IStringInstance Interface [▶ 1608]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.58 IStringType Interface

Interface representing a string IDataType [▶ 1517]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IStringType : IDataType,
    IBitSize
```

VB

```
Public Interface IStringType
    Inherits IDataType, IBitSize
```

The IStringType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	Encoding [▶ 1614]	Gets the encoding of the String (Encoding.Default (STRING) or Encoding.UNICODE (WSTRING))
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)

	Name	Description
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsFixedLength [▶ 1615]	Gets a value indicating whether the string is of fixed length.
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Length [▶ 1615]	Gets the number of characters within the string (when fixed length).
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.58.1 IStringType Properties

The IStringType [▶ 1612] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	Encoding [▶ 1614]	Gets the encoding of the String (Encoding.Default (STRING) or Encoding.UNICODE (WSTRING))
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)

	Name	Description
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsFixedLength [▶ 1615]	Gets a value indicating whether the string is of fixed length.
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Length [▶ 1615]	Gets the number of characters within the string (when fixed length).
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

IStringType Interface [▶ 1612]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.58.1.1 IStringType.Encoding Property

Gets the encoding of the String (Encoding.Default (STRING) or Encoding.UNICODE (WSTRING))

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Encoding Encoding { get; }
```

VB

```
ReadOnly Property Encoding As Encoding
    Get
```

Property Value

Type: Encoding
The encoding.

Reference

IStringType Interface [▶ 1612]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.58.1.2 IStringType.IsFixedLength Property

Gets a value indicating whether the string is of fixed length.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsFixedLength { get; }
```

VB

```
ReadOnly Property IsFixedLength As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is fixed length; otherwise, false.

Reference

IStringType Interface [▶ 1612]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.58.1.3 IStringType.Length Property

Gets the number of characters within the string (when fixed length).

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int Length { get; }
```

VB

```
ReadOnly Property Length As Integer
    Get
```

Property Value

Type: Int32
The length if fixed length, otherwise -1

Reference

IStringType Interface [▶ 1612]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.59 IStructInstance Interface

Interface representing an instance of a IStructType [▶ 1619]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IStructInstance : ISymbol,
    IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IStructInstance
    Inherits ISymbol, IAttributedInstance, IInstance, IBitSize
```

The IStructInstance type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasRpcMethods [▶ 1618]	Gets a value indicating whether this instance has RPC methods
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)

	Name	Description
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	MemberInstances [▶ 1619]	Gets the member instances of the Struct Instance.
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.59.1 IStructInstance Properties

The IStructInstance [▶ 1615] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasRpcMethods [▶ 1618]	Gets a value indicating whether this instance has RPC methods
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)

	Name	Description
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	MemberInstances [▶ 1619]	Gets the member instances of the Struct Instance [▶ 1615].
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IStructInstance Interface [▶ 1615]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.59.1.1 IStructInstance.HasRpcMethods Property

Gets a value indicating whether this instance has RPC methods

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool HasRpcMethods { get; }
```

VB

```
ReadOnly Property HasRpcMethods As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has RPC methods; otherwise, false.

Remarks

If the struct instance supports RPC Methods, then the instance class is also supporting IRpcStructInstance [▶ 1604]:

Reference

IStructInstance Interface [▶ 1615]

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IRpcStructInstance [▶ 1604]

TwinCAT.TypeSystem.IRpcMethod [▶ 1597]

TwinCAT.TypeSystem.IRpcMethodParameter [▶ 1601]

5.8.59.1.2 IStructInstance.MemberInstances Property

Gets the member instances of the Struct Instance [▶ 1615].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlySymbolCollection MemberInstances { get; }
```

VB

```
ReadOnly Property MemberInstances As ReadOnlySymbolCollection
    Get
```

Property Value

Type: ReadOnlySymbolCollection [▶ 1772]
The member instances.

Reference

IStructInstance Interface [▶ 1615]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.60 IStructType Interface

Interface representing Struct data types

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IStructType : IDataType,
    IBitSize
```

VB

```
Public Interface IStructType
    Inherits IDataType, IBitSize
```

The IStructType type exposes the following members.

Properties

	Name	Description
	AllMembers [▶ 1622]	Gets all members (down the derivation hierarchy)
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 1622]	Gets the structs Base Type (Null if not derived).
	BaseTypeName [▶ 1623]	Gets the the Name of the Base class (if derived)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	HasRpcMethods [▶ 1623]	Gets a value indicating whether this instance has RPC methods (Struct types only)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Members [▶ 1624]	Gets a readonly collection of the Members [▶ 1562] of the IStructType.
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)

	Name	Description
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.60.1 IStructType Properties

The IStructType [▶ 1619] type exposes the following members.

Properties

	Name	Description
	AllMembers [▶ 1622]	Gets all members (down the derivation hierarchy)
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 1622]	Gets the structs Base Type (Null if not derived).
	BaseTypeName [▶ 1623]	Gets the the Name of the Base class (if derived)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	HasRpcMethods [▶ 1623]	Gets a value indicating whether this instance has RPC methods (Struct types only)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)

	Name	Description
	Members [▶ 1624]	Gets a readonly collection of the Members [▶ 1562] of the IStructType [▶ 1619].
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

IStructType Interface [▶ 1619]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.60.1.1 IStructType.AllMembers Property

Gets all members (down the derivation hierarchy)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyMemberCollection AllMembers { get; }
```

VB

```
ReadOnly Property AllMembers As ReadOnlyMemberCollection
    Get
```

Property Value

Type: ReadOnlyMemberCollection [▶ 1755]
All members.

Reference

IStructType Interface [▶ 1619]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.60.1.2 IStructType.BaseType Property

Gets the structs Base Type (Null if not derived).

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType BaseType { get; }
```

VB

```
ReadOnly Property BaseType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]

Reference

IStructType Interface [▶ 1619]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.60.1.3 IStructType.BaseTypeName Property

Gets the the Name of the Base class (if derived)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string BaseTypeName { get; }
```

VB

```
ReadOnly Property BaseTypeName As String
    Get
```

Property Value

Type: String
Empty if not derived.

Reference

IStructType Interface [▶ 1619]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.60.1.4 IStructType.HasRpcMethods Property

Gets a value indicating whether this instance has RPC methods (Struct types only)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool HasRpcMethods { get; }
```

VB

```
ReadOnly Property HasRpcMethods As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has RPC methods; otherwise, false.

Remarks

The DataType (Structure) must be marked with the PlcAttribute 'TcRpcEnable' to enable RpcMethods, otherwise RpcMethods are not passed through to the ADS symbolic information.

Reference

IStructType Interface [▶ 1619]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.60.1.5 IStructType.Members Property

Gets a readonly collection of the Members [▶ 1562] of the IStructType [▶ 1619].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyMemberCollection Members { get; }
```

VB

```
ReadOnly Property Members As ReadOnlyMemberCollection
    Get
```

Property Value

Type: ReadOnlyMemberCollection [▶ 1755]
The members as readonly collection.

Remarks

If the IStructType [▶ 1619] is derived, only the extended members are returned. To get all supported members down the inheritance chain, use the AllMembers [▶ 1622] property.

Reference

IStructType Interface [▶ 1619]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.61 IStructValue Interface

Interface IStructValue

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IStructValue : IValue
```

VB

```
Public Interface IStructValue
    Inherits IValue
```

The IStructValue type exposes the following members.

Properties

	Name	Description
	Age [▶ 1660]	Gets the age of the value (last successful read of the value) (Inherited from IValue [▶ 1659].)
	CachedRaw [▶ 1661]	Gets the cached Raw internal Data. (Inherited from IValue [▶ 1659].)
	DataType [▶ 1661]	Gets the data type bound to this IValue [▶ 1659] (Inherited from IValue [▶ 1659].)
	IsPrimitive [▶ 1662]	Gets a value indicating whether this IValue [▶ 1659] is a primitive value. (Inherited from IValue [▶ 1659].)
	Symbol [▶ 1662]	Gets the symbol bound to this IValue [▶ 1659]. (Inherited from IValue [▶ 1659].)
	UtcTimeStamp [▶ 1663]	Gets the Time stamp of the last successful read of the Value (local user time, UTC) (Inherited from IValue [▶ 1659].)

Methods

	Name	Description
	Read [▶ 1664]	Reads the value (via ADS) (Inherited from IValue [▶ 1659].)
	ResolveValue [▶ 1664]	Resolves the Value object to its primitive value. (Inherited from IValue [▶ 1659].)
	TryGetMemberValue [▶ 1626]	Tries to get a property/Member value.
	TryResolveValue [▶ 1665]	Tries to resolves the Value object to its primitive value. (Inherited from IValue [▶ 1659].)
	TrySetMemberValue [▶ 1627]	Tries to Set a Member/Property Value
	Write [▶ 1665]	Writes the value (via ADS) (Inherited from IValue [▶ 1659].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IValue [▶ 1659]

5.8.61.1 IStructValue Properties

The IStructValue [▶ 1624] type exposes the following members.

Properties

	Name	Description
	Age [▶ 1660]	Gets the age of the value (last successful read of the value) (Inherited from IValue [▶ 1659].)
	CachedRaw [▶ 1661]	Gets the cached Raw internal Data. (Inherited from IValue [▶ 1659].)
	DataType [▶ 1661]	Gets the data type bound to this IValue [▶ 1659] (Inherited from IValue [▶ 1659].)
	IsPrimitive [▶ 1662]	Gets a value indicating whether this IValue [▶ 1659] is a primitive value. (Inherited from IValue [▶ 1659].)
	Symbol [▶ 1662]	Gets the symbol bound to this IValue [▶ 1659]. (Inherited from IValue [▶ 1659].)
	UtcTimeStamp [▶ 1663]	Gets the Time stamp of the last successful read of the Value (local user time, UTC) (Inherited from IValue [▶ 1659].)

Reference

IStructValue Interface [▶ 1624]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.61.2 IStructValue Methods

The IStructValue [▶ 1624] type exposes the following members.

Methods

	Name	Description
	Read [▶ 1664]	Reads the value (via ADS) (Inherited from IValue [▶ 1659].)
	ResolveValue [▶ 1664]	Resolves the Value object to its primitive value. (Inherited from IValue [▶ 1659].)
	TryGetMemberValue [▶ 1626]	Tries to get a property/Member value.
	TryResolveValue [▶ 1665]	Tries to resolves the Value object to its primitive value. (Inherited from IValue [▶ 1659].)
	TrySetMemberValue [▶ 1627]	Tries to Set a Member/Property Value
	Write [▶ 1665]	Writes the value (via ADS) (Inherited from IValue [▶ 1659].)

Reference

IStructValue Interface [▶ 1624]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.61.2.1 IStructValue.TryGetMemberValue Method

Tries to get a property/Member value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TryGetMemberValue(
    string name,
    out Object value
)
```

VB

```
Function TryGetMemberValue (
    name As String,
    <OutAttribute> ByRef value As Object
) As Boolean
```

Parameters

name	Type: System.String The name of the member
value	Type: System.Object. The value.

Return Value

Type: Boolean
true if succeeded, otherwise false otherwise.

Reference

IStructValue Interface [▶ 1624]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.61.2.2 IStructValue.TrySetMemberValue Method

Tries to Set a Member/Property Value

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TrySetMemberValue(
    string name,
    Object value
)
```

VB

```
Function TrySetMemberValue (
    name As String,
    value As Object
) As Boolean
```

Parameters

name	Type: System.String The name of the member
value	Type: System.Object The value.

Return Value

Type: Boolean
true if succeeded, otherwise false otherwise.

Reference

IStructValue Interface [▶ 1624]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.62 ISubRangeType Interface

Interface representing a SubRange type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISubRangeType : IDataType,
    IBitSize
```

VB

```
Public Interface ISubRangeType
    Inherits IDataType, IBitSize
```

The ISubRangeType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 1630]	Gets the the base type.
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)

	Name	Description
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.62.1 ISubRangeType Properties

The ISubRangeType [▶ 1628] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 1630]	Gets the the base type.
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)

	Name	Description
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

ISubRangeType Interface [▶ 1628]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.62.1.1 ISubRangeType.BaseType Property

Gets the the base type.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType BaseType { get; }
```

VB

```
ReadOnly Property BaseType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the referenced.

Reference

ISubRangeType Interface [▶ 1628]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.63 ISubRangeType.T. Interface

Interface representing a SubRange type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISubRangeType<T> : ISubRangeType,
    IDataType, IBitSize
where T : struct, new()
```

VB

```
Public Interface ISubRangeType(Of T As {Structure, New})
    Inherits ISubRangeType, IDataType, IBitSize
```

Type Parameters

T

The ISubRangeType.T. type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 1630]	Gets the the base type. (Inherited from ISubRangeType [▶ 1628].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	LowerBound [▶ 1633]	Gets the lower bound.
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)

	Name	Description
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	UpperBound [▶ 1633]	Gets the upper bound.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.63.1 ISubRangeType.T. Properties

The ISubRangeType.T. [▶ 1630] generic type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BaseType [▶ 1630]	Gets the the base type. (Inherited from ISubRangeType [▶ 1628].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	LowerBound [▶ 1633]	Gets the lower bound.
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)

	Name	Description
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	UpperBound [▶ 1633]	Gets the upper bound.

Reference

ISubRangeType.T. Interface [▶ 1630]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.63.1.1 ISubRangeType.T..LowerBound Property

Gets the lower bound.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
T LowerBound { get; }
```

VB

```
ReadOnly Property LowerBound As T
    Get
```

Property Value

Type: T [▶ 1630]
The lower bound.

Reference

ISubRangeType.T. Interface [▶ 1630]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.63.1.2 ISubRangeType.T..UpperBound Property

Gets the upper bound.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
T UpperBound { get; }
```

VB

```
ReadOnly Property UpperBound As T
    Get
```

Property Value

Type: T [▶ 1630]
The upper bound.

Reference

ISubRangeType.T. Interface [▶ 1630]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.64 ISymbol Interface

Interface specifying Symbols (

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISymbol : IAttributedInstance,
    IInstance, IBitSize
```

VB

```
Public Interface ISymbol
    Inherits IAttributedInstance, IInstance, IBitSize
```

The ISymbol type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)

	Name	Description
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type.
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol is persistent.
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type.
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol is read only.
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive.
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.64.1 ISymbol Properties

The ISymbol [▶ 1634] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)

	Name	Description
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type.
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent.
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type.
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only.
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive.
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634]
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

ISymbol Interface [▶ 1634]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.64.1.1 ISymbol.Category Property

Gets the Symbol/Datatype Category

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
DataTypeCategory Category { get; }
```

VB

```
ReadOnly Property Category As DataTypeCategory
    Get
```

Property Value

Type: DataTypeCategory [▶ 1189]
The category.

Reference

ISymbol Interface [▶ 1634]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.64.1.2 ISymbol.IsContainerType Property

Gets a value indicating whether this Symbol is a container type.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsContainerType { get; }
```

VB

```
ReadOnly Property IsContainerType As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is container type; otherwise, false.

Reference

ISymbol Interface [▶ 1634]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.64.1.3 ISymbol.IsPersistent Property

Gets a value indicating whether this ISymbol [▶ 1634] is persistent.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsPersistent { get; }
```

VB

```
ReadOnly Property IsPersistent As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is persistent; otherwise, false.

Reference

ISymbol Interface [▶ 1634]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.64.1.4 ISymbol.IsPrimitiveType Property

Gets a value indicating whether this instance is a primitive type.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsPrimitiveType { get; }
```

VB

```
ReadOnly Property IsPrimitiveType As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is primitive type; otherwise, false.

Reference

ISymbol Interface [▶ 1634]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.64.1.5 ISymbol.IsReadOnly Property

Gets a value indicating whether this ISymbol [▶ 1634] is read only.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsReadOnly { get; }
```

VB

```
ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is read only; otherwise, false.

Reference

ISymbol Interface [▶ 1634]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.64.1.6 ISymbol.IsRecursive Property

Gets a value indicating whether this instance is recursive.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsRecursive { get; }
```

VB

```
ReadOnly Property IsRecursive As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is recursive; otherwise, false.

Reference

ISymbol Interface [▶ 1634]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.64.1.7 ISymbol.Parent Property

Gets the parent Symbol

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ISymbol Parent { get; }
```

VB

```
ReadOnly Property Parent As ISymbol
    Get
```

Property Value

Type: ISymbol [▶ 1634]
The parent.

Reference

ISymbol Interface [▶ 1634]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.64.1.8 ISymbol.SubSymbols Property

Gets the SubSymbols of the ISymbol [▶ 1634]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlySymbolCollection SubSymbols { get; }
```

VB

```
ReadOnly Property SubSymbols As ReadOnlySymbolCollection
    Get
```

Property Value

Type: ReadOnlySymbolCollection [▶ 1772]

Remarks

Used for Array, Struct, Pointer and Reference instances. Otherwise empty

Reference

ISymbol Interface [▶ 1634]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.65 ISymbolCollection Interface

Interface ISymbolCollection

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISymbolCollection : ISymbolCollection<ISymbol>,
    IInstanceCollection<ISymbol>, IList<ISymbol>, ICollection<ISymbol>,
    IEnumerable<ISymbol>, IEnumerable
```

VB

```
Public Interface ISymbolCollection
    Inherits ISymbolCollection(Of ISymbol), IInstanceCollection(Of ISymbol),
    IList(Of ISymbol), ICollection(Of ISymbol), IEnumerable(Of ISymbol),
    IEnumerable
```

The ISymbolCollection type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ICollection.T.. (Inherited from ICollection.ISymbol [▶ 1634]..)
	IsReadOnly	Gets a value indicating whether the ICollection.T. is read-only. (Inherited from ICollection.ISymbol [▶ 1634]..)
	Item.Int32.	Gets or sets the element at the specified index. (Inherited from IList.ISymbol [▶ 1634]..)
	Item.String. [▶ 1879]	Gets the IInstance [▶ 1556] with the specified instance path. (Inherited from IInstanceCollection.T. [▶ 1877].)
	Mode [▶ 1880]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from IInstanceCollection.T. [▶ 1877].)

Methods

	Name	Description
	Add	Adds an item to the ICollection.T.. (Inherited from ICollection.ISymbol [▶ 1634]..)
	Clear	Removes all items from the ICollection.T.. (Inherited from ICollection.ISymbol [▶ 1634]..)
	Contains(String) [▶ 1881]	Determines whether this collection contains an instance with the specified instance path. (Inherited from IInstanceCollection.T. [▶ 1877].)
	Contains(T)	Determines whether the ICollection.T. contains a specific value. (Inherited from ICollection.ISymbol [▶ 1634]..)
	ContainsName [▶ 1882]	Determines whether this collection contains an instance with the specified instance name. (Inherited from IInstanceCollection.T. [▶ 1877].)
	CopyTo	Copies the elements of the ICollection.T. to an Array, starting at a particular Array index. (Inherited from ICollection.ISymbol [▶ 1634]..)
	GetEnumerator	Returns an enumerator that iterates through the collection. (Inherited from IEnumerable.ISymbol [▶ 1634]..)
	GetInstance [▶ 1883]	Gets the IInstance [▶ 1556]by instance path. (Inherited from IInstanceCollection.T. [▶ 1877].)
	GetInstanceByName [▶ 1883]	Gets the IInstance [▶ 1556] by instance name. (Inherited from IInstanceCollection.T. [▶ 1877].)
	IndexOf	Determines the index of a specific item in the IList.T.. (Inherited from IList.ISymbol [▶ 1634]..)
	Insert	Inserts an item to the IList.T. at the specified index. (Inherited from IList.ISymbol [▶ 1634]..)
	Remove	Removes the first occurrence of a specific object from the ICollection.T.. (Inherited from ICollection.ISymbol [▶ 1634]..)
	RemoveAt	Removes the IList.T. item at the specified index. (Inherited from IList.ISymbol [▶ 1634]..)

	Name	Description
	TryGetInstance [▶ 1884]	Tries to get the specified instance. (Inherited from IInstanceCollection.T. [▶ 1877].)
	TryGetInstanceByName [▶ 1884]	Tries to get the specified instance by name. (Inherited from IInstanceCollection.T. [▶ 1877].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.65.1 ISymbolCollection Properties

The ISymbolCollection [▶ 1640] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ICollection.T.. (Inherited from ICollection.ISymbol [▶ 1634]..)
	IsReadOnly	Gets a value indicating whether the ICollection.T. is read-only. (Inherited from ICollection.ISymbol [▶ 1634]..)
	Item.Int32.	Gets or sets the element at the specified index. (Inherited from IList.ISymbol [▶ 1634]..)
	Item.String. [▶ 1879]	Gets the IInstance [▶ 1556] with the specified instance path. (Inherited from IInstanceCollection.T. [▶ 1877].)
	Mode [▶ 1880]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from IInstanceCollection.T. [▶ 1877].)

Reference

ISymbolCollection Interface [▶ 1640]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.65.2 ISymbolCollection Methods

The ISymbolCollection [▶ 1640] type exposes the following members.

Methods

	Name	Description
	Add	Adds an item to the ICollection.T.. (Inherited from ICollection.ISymbol [▶ 1634]..)
	Clear	Removes all items from the ICollection.T.. (Inherited from ICollection.ISymbol [▶ 1634]..)
	Contains(String) [▶ 1881]	Determines whether this collection contains an instance with the specified instance path. (Inherited from IInstanceCollection.T. [▶ 1877].)
	Contains(T)	Determines whether the ICollection.T. contains a specific value. (Inherited from ICollection.ISymbol [▶ 1634]..)
	ContainsName [▶ 1882]	Determines whether this collection contains an instance with the specified instance name. (Inherited from IInstanceCollection.T. [▶ 1877].)

	Name	Description
	CopyTo	Copies the elements of the ICollection.T. to an Array, starting at a particular Array index. (Inherited from ICollection.ISymbol [▶ 1634]..)
	GetEnumerator	Returns an enumerator that iterates through the collection. (Inherited from IEnumerable.ISymbol [▶ 1634]..)
	GetInstance [▶ 1883]	Gets the IInstance [▶ 1556]by instance path. (Inherited from IInstanceCollection.T. [▶ 1877].)
	GetInstanceByName [▶ 1883]	Gets the IInstance [▶ 1556] by instance name. (Inherited from IInstanceCollection.T. [▶ 1877].)
	IndexOf	Determines the index of a specific item in the IList.T.. (Inherited from IList.ISymbol [▶ 1634]..)
	Insert	Inserts an item to the IList.T. at the specified index. (Inherited from IList.ISymbol [▶ 1634]..)
	Remove	Removes the first occurrence of a specific object from the ICollection.T.. (Inherited from ICollection.ISymbol [▶ 1634]..)
	RemoveAt	Removes the IList.T. item at the specified index. (Inherited from IList.ISymbol [▶ 1634]..)
	TryGetInstance [▶ 1884]	Tries to get the specified instance. (Inherited from IInstanceCollection.T. [▶ 1877].)
	TryGetInstanceByName [▶ 1884]	Tries to get the specified instance by name. (Inherited from IInstanceCollection.T. [▶ 1877].)

Reference

ISymbolCollection Interface [▶ 1640]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.66 ISymbolInfo Interface

Interface ISymbolInfo

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISymbolInfo
```

VB

```
Public Interface ISymbolInfo
```

The ISymbolInfo type exposes the following members.

Properties

	Name	Description
	InstancePath [▶ 1644]	Gets the Symbol Path
	TypeName [▶ 1644]	Gets the data type Name

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.66.1 ISymbolInfo Properties

The ISymbolInfo [▶ 1643] type exposes the following members.

Properties

	Name	Description
	InstancePath [▶ 1644]	Gets the Symbol Path
	TypeName [▶ 1644]	Gets the data type Name

Reference

ISymbolInfo Interface [▶ 1643]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.66.1.1 ISymbolInfo.InstancePath Property

Gets the Symbol Path

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string InstancePath { get; }
```

VB

```
ReadOnly Property InstancePath As String
    Get
```

Property Value

Type: String
The path.

Reference

ISymbolInfo Interface [▶ 1643]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.66.1.2 ISymbolInfo.TypeName Property

Gets the data type Name

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string TypeName { get; }
```

VB

```
ReadOnly Property TypeName As String
    Get
```

Property Value

Type: String
The type of the data.

Reference

ISymbolInfo Interface [▶ 1643]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.67 ISymbolLoader Interface

Symbol Loader interface

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISymbolLoader : ISymbolProvider
```

VB

```
Public Interface ISymbolLoader
    Inherits ISymbolProvider
```

The ISymbolLoader type exposes the following members.

Properties

	Name	Description
	BuildInTypes [▶ 1646]	Gets the build in types.
	DataTypes [▶ 1648]	Gets all data types from all Namespaces (Inherited from ISymbolProvider [▶ 1647].)
	RootNamespaceName [▶ 1648]	Gets the name of the root namespace (Inherited from ISymbolProvider [▶ 1647].)
	Settings [▶ 1646]	Gets or sets the access Method
	Symbols [▶ 1649]	Gets the (root) symbols of the Symbol provider. (Inherited from ISymbolProvider [▶ 1647].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.67.1 ISymbolLoader Properties

The ISymbolLoader [▶ 1645] type exposes the following members.

Properties

	Name	Description
	BuildInTypes [▶ 1646]	Gets the build in types.
	DataTypes [▶ 1648]	Gets all data types from all Namespaces (Inherited from ISymbolProvider [▶ 1647].)
	RootNamespaceName [▶ 1648]	Gets the name of the root namespace (Inherited from ISymbolProvider [▶ 1647].)
	Settings [▶ 1646]	Gets or sets the access Method
	Symbols [▶ 1649]	Gets the (root) symbols of the Symbol provider. (Inherited from ISymbolProvider [▶ 1647].)

Reference

ISymbolLoader Interface [▶ 1645]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.67.1.1 ISymbolLoader.BuildInTypes Property

Gets the build in types.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyDataTypeCollection<IDataType> BuildInTypes { get; }
```

VB

```
ReadOnly Property BuildInTypes As ReadOnlyDataTypeCollection(Of IDataType)
    Get
```

Property Value

Type: ReadOnlyDataTypeCollection [▶ 1935].IDataType [▶ 1517].
The build in types.

Reference

ISymbolLoader Interface [▶ 1645]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.67.1.2 ISymbolLoader.Settings Property

Gets or sets the access Method

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ISymbolLoaderSettings Settings { get; }
```

VB

```
ReadOnly Property Settings As ISymbolLoaderSettings
    Get
```

Property Value

Type: ISymbolLoaderSettings [▶ 66]
The access method.

Reference

ISymbolLoader Interface [▶ 1645]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.68 ISymbolProvider Interface

Symbol Provider interface.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISymbolProvider
```

VB

```
Public Interface ISymbolProvider
```

The ISymbolProvider type exposes the following members.

Properties

	Name	Description
	DataTypes [▶ 1648]	Gets all data types from all Namespaces
	RootNamespaceName [▶ 1648]	Gets the name of the root namespace
	Symbols [▶ 1649]	Gets the (root) symbols of the Symbol provider.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.68.1 ISymbolProvider Properties

The ISymbolProvider [▶ 1647] type exposes the following members.

Properties

	Name	Description
	DataTypes [▶ 1648]	Gets all data types from all Namespaces
	RootNamespaceName [▶ 1648]	Gets the name of the root namespace
	Symbols [▶ 1649]	Gets the (root) symbols of the Symbol provider.

Reference

ISymbolProvider Interface [▶ 1647]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.68.1.1 ISymbolProvider.DataTypes Property

Gets all data types from all Namespaces

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyDataTypeCollection DataTypes { get; }
```

VB

```
ReadOnly Property DataTypes As ReadOnlyDataTypeCollection
    Get
```

Property Value

Type: ReadOnlyDataTypeCollection [▶ 1727]
The data types.

Reference

ISymbolProvider Interface [▶ 1647]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.68.1.2 ISymbolProvider.RootNamespaceName Property

Gets the name of the root namespace

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string RootNamespaceName { get; }
```

VB

```
ReadOnly Property RootNamespaceName As String
    Get
```

Property Value

Type: String
The namespace.

Reference

ISymbolProvider Interface [▶ 1647]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.68.1.3 ISymbolProvider.Symbols Property

Gets the (root) symbols of the Symbol provider.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlySymbolCollection Symbols { get; }
```

VB

```
ReadOnly Property Symbols As ReadOnlySymbolCollection
    Get
```

Property Value

Type: ReadOnlySymbolCollection [▶ 1772]
Read only collection of the Symbols

Reference

ISymbolProvider Interface [▶ 1647]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.69 ISymbolServer Interface

Symbol Server Interface

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISymbolServer
```

VB

```
Public Interface ISymbolServer
```

The ISymbolServer type exposes the following members.

Properties

	Name	Description
	DataTypes [▶ 1650]	Gets the data types.
	Symbols [▶ 1651]	Gets the symbols.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.69.1 ISymbolServer Properties

The ISymbolServer [▶ 1649] type exposes the following members.

Properties

	Name	Description
	DataTypes [▶ 1650]	Gets the data types.
	Symbols [▶ 1651]	Gets the symbols.

Reference

ISymbolServer Interface [▶ 1649]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.69.1.1 ISymbolServer.DataTypes Property

Gets the data types.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyDataTypeCollection DataTypes { get; }
```

VB

```
ReadOnly Property DataTypes As ReadOnlyDataTypeCollection
    Get
```

Property Value

Type: ReadOnlyDataTypeCollection [▶ 1727]
The data types.

Reference

ISymbolServer Interface [▶ 1649]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.69.1.2 ISymbolServer.Symbols Property

Gets the symbols.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlySymbolCollection Symbols { get; }
```

VB

```
ReadOnly Property Symbols As ReadOnlySymbolCollection
    Get
```

Property Value

Type: ReadOnlySymbolCollection [▶ 1772]
The symbols.

Reference

ISymbolServer Interface [▶ 1649]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.70 ITypeAttribute Interface

Interface for ADS attributes

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ITypeAttribute
```

VB

```
Public Interface ITypeAttribute
```

The ITypeAttribute type exposes the following members.

Properties

	Name	Description
	Name [▶ 1652]	Name of the Attribute
	Value [▶ 1653]	Gets the value of the attribute

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.70.1 ITypeAttribute Properties

The ITypeAttribute [▶ 1651] type exposes the following members.

Properties

	Name	Description
	Name [▶ 1652]	Name of the Attribute
	Value [▶ 1653]	Gets the value of the attribute

Reference

ITypeAttribute Interface [▶ 1651]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.70.1.1 ITypeAttribute.Name Property

Name of the Attribute

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Name { get; }
```

VB

```
ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Reference

ITypeAttribute Interface [▶ 1651]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.70.1.2 ITypeAttribute.Value Property

Gets the value of the attribute

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Value { get; }
```

VB

```
ReadOnly Property Value As String
    Get
```

Property Value

Type: String
The value.

Reference

ITypeAttribute Interface [▶ 1651]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.71 IUnionInstance Interface

Interface for an Instance of the IUnionType [▶ 1656].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IUnionInstance : ISymbol,
    IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IUnionInstance
    Inherits ISymbol, IAttributedInstance, IInstance, IBitSize
```

The IUnionInstance type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)

	Name	Description
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	FieldInstances [▶ 1656]	Gets the field instances of the Union
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.ISymbol [▶ 1634]

5.8.71.1 IUnionInstance Properties

The IUnionInstance [▶ 1653] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	FieldInstances [▶ 1656]	Gets the field instances of the Union
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IUnionInstance Interface [▶ 1653]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.71.1.1 IUnionInstance.FieldInstances Property

Gets the field instances of the Union

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlySymbolCollection FieldInstances { get; }
```

VB

```
ReadOnly Property FieldInstances As ReadOnlySymbolCollection
    Get
```

Property Value

Type: ReadOnlySymbolCollection [▶ 1772]
The field instances.

Reference

IUnionInstance Interface [▶ 1653]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.72 IUnionType Interface

Interface for an union data type.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IUnionType : IDataType,
    IBitSize
```

VB

```
Public Interface IUnionType
    Inherits IDataType, IBitSize
```

The IUnionType type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)

	Name	Description
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	Fields [▶ 1658]	Gets a readonly collection of the Members [▶ 1554] of the IUnionType.
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IDataType [▶ 1517]

5.8.72.1 IUnionType Properties

The IUnionType [▶ 1656] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1519]	Gets the attributes of the IDataType [▶ 1517] (Inherited from IDataType [▶ 1517].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)

	Name	Description
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1520]	Gets the Data Type category (Inherited from IDataType [▶ 1517].)
	Comment [▶ 1520]	Gets the comment behind the variable declaration. (Inherited from IDataType [▶ 1517].)
	Fields [▶ 1658]	Gets a readonly collection of the Members [▶ 1554] of the IUnionType [▶ 1656].
	FullName [▶ 1521]	Gets the full name of the IDataType [▶ 1517] (Namespace + Name) (Inherited from IDataType [▶ 1517].)
	Id [▶ 1521]	Gets the ID of the DataType (Inherited from IDataType [▶ 1517].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainer [▶ 1522]	Gets a value indicating whether this IDataType [▶ 1517] is a container type (Inherited from IDataType [▶ 1517].)
	IsPointer [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is a pointer type (Inherited from IDataType [▶ 1517].)
	IsPrimitive [▶ 1523]	Gets a value indicating whether this IDataType [▶ 1517] is primitive (Inherited from IDataType [▶ 1517].)
	IsReference [▶ 1524]	Gets a value indicating whether this IDataType [▶ 1517] is a reference type (Inherited from IDataType [▶ 1517].)
	Name [▶ 1524]	Gets the name of the Data Type (without namespace) (Inherited from IDataType [▶ 1517].)
	Namespace [▶ 1525]	Gets the namespace string within the IDataType [▶ 1517] exists. (Inherited from IDataType [▶ 1517].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

Reference

IUnionType Interface [▶ 1656]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.72.1.1 IUnionType.Fields Property

Gets a readonly collection of the Members [▶ 1554] of the IUnionType [▶ 1656].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyFieldCollection Fields { get; }
```

VB

```
ReadOnly Property Fields As ReadOnlyFieldCollection
    Get
```

Property Value

Type: ReadOnlyFieldCollection [▶ 1750]
The members as readonly collection.

Remarks

If the IStructType [▶ 1619] is derived, only the extended members are returned. To get all supported members down the inheritance chain, use the AllMembers [▶ 1622] property.

Reference

IUnionType Interface [▶ 1656]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.73 IValue Interface

Symbol Value Interface

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IValue
```

VB

```
Public Interface IValue
```

The IValue type exposes the following members.

Properties

	Name	Description
	Age [▶ 1660]	Gets the age of the value (last successful read of the value)
	CachedRaw [▶ 1661]	Gets the cached Raw internal Data.
	DataType [▶ 1661]	Gets the data type bound to this IValue
	IsPrimitive [▶ 1662]	Gets a value indicating whether this IValue is a primitive value.
	Symbol [▶ 1662]	Gets the symbol bound to this IValue.
	UtcTimeStamp [▶ 1663]	Gets the Time stamp of the last successful read of the Value (local user time, UTC)

Methods

	Name	Description
	Read [▶ 1664]	Reads the value (via ADS)

	Name	Description
	ResolveValue [▶ 1664]	Resolves the Value object to its primitive value.
	TryResolveValue [▶ 1665]	Tries to resolves the Value object to its primitive value.
	Write [▶ 1665]	Writes the value (via ADS)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.73.1 IValue Properties

The IValue [▶ 1659] type exposes the following members.

Properties

	Name	Description
	Age [▶ 1660]	Gets the age of the value (last successful read of the value)
	CachedRaw [▶ 1661]	Gets the cached Raw internal Data.
	DataType [▶ 1661]	Gets the data type bound to this IValue [▶ 1659]
	IsPrimitive [▶ 1662]	Gets a value indicating whether this IValue [▶ 1659] is a primitive value.
	Symbol [▶ 1662]	Gets the symbol bound to this IValue [▶ 1659].
	UtcTimeStamp [▶ 1663]	Gets the Time stamp of the last successful read of the Value (local user time, UTC)

Reference

IValue Interface [▶ 1659]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.73.1.1 IValue.Age Property

Gets the age of the value (last successful read of the value)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
TimeSpan Age { get; }
```

VB

```
ReadOnly Property Age As TimeSpan
    Get
```

Property Value

Type: TimeSpan
The age.

Reference

IValue Interface [▶ 1659]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValue.UtcTimeStamp [▶ 1663]

5.8.73.1.2 IValue.CachedRaw Property

Gets the cached Raw internal Data.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
byte[] CachedRaw { get; }
```

VB

```
ReadOnly Property CachedRaw As Byte()
    Get
```

Property Value

Type: .Byte.
The raw cached data.

Reference

IValue Interface [▶ 1659]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.73.1.3 IValue.DataType Property

Gets the data type bound to this IValue [▶ 1659]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IDataType DataType { get; }
```

VB

```
ReadOnly Property DataType As IDataType
    Get
```

Property Value

Type: IDataType [▶ 1517]
The type of the data.

Reference

IValue Interface [▶ 1659]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.73.1.4 IValue.IsPrimitive Property

Gets a value indicating whether this IValue [▶ 1659] is a primitive value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool IsPrimitive { get; }
```

VB

```
ReadOnly Property IsPrimitive As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is primitive; otherwise, false.

Reference

IValue Interface [▶ 1659]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.73.1.5 IValue.Symbol Property

Gets the symbol bound to this IValue [▶ 1659].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ISymbol Symbol { get; }
```

VB

```
ReadOnly Property Symbol As ISymbol
    Get
```

Property Value

Type: ISymbol [▶ 1634]
The symbol.

Reference

IValue Interface [▶ 1659]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.73.1.6 IValue.UtcTimeStamp Property

Gets the Time stamp of the last successful read of the Value (local user time, UTC)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
DateTime UtcTimeStamp { get; }
```

VB

```
ReadOnly Property UtcTimeStamp As DateTime
    Get
```

Property Value

Type: DateTime
The read time stamp.

Reference

IValue Interface [▶ 1659]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.73.2 IValue Methods

The IValue [▶ 1659] type exposes the following members.

Methods

	Name	Description
	Read [▶ 1664]	Reads the value (via ADS)
	ResolveValue [▶ 1664]	Resolves the Value object to its primitive value.
	TryResolveValue [▶ 1665]	Tries to resolves the Value object to its primitive value.
	Write [▶ 1665]	Writes the value (via ADS)

Reference

IValue Interface [▶ 1659]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.73.2.1 IValue.Read Method

Reads the value (via ADS)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void Read()
```

VB

```
Sub Read
```

Reference

IValue Interface [▶ 1659]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.73.2.2 IValue.ResolveValue Method

Resolves the Value object to its primitive value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object ResolveValue(
    bool resolveEnumToPrimitive
)
```

VB

```
Function ResolveValue (
    resolveEnumToPrimitive As Boolean
) As Object
```

Parameters

resolveEnumToPrimitive	Type: System.Boolean if set to trueIEnumValue [▶ 1550]s are resolved to their primitives also.

Return Value

Type: Object
System.Object.

Remarks

If the value is not primitive, this method returns the IValue [▶ 1659] itself.

Reference

IValue Interface [▶ 1659]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.73.2.3 IValue.TryResolveValue Method

Tries to resolves the Value object to its primitive value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TryResolveValue(
    bool resolveEnumToPrimitive,
    out Object value
)
```

VB

```
Function TryResolveValue (
    resolveEnumToPrimitive As Boolean,
    <OutAttribute> ByRef value As Object
) As Boolean
```

Parameters

resolveEnumToPrimitive	Type: System.Boolean if set to trueIEnumValue [▶ 1550]s are resolved to their primitives also.
value	Type: System.Object. The value.

Return Value

Type: Boolean
true if value can be resolved, false otherwise.

Reference

IValue Interface [▶ 1659]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.73.2.4 IValue.Write Method

Writes the value (via ADS)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void Write()
```

VB

```
Sub Write
```

Reference

IValue Interface [▶ 1659]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.74 IValueAnySymbol Interface

Interface IValueAnySymbol

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IValueAnySymbol : ISymbol,
    IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IValueAnySymbol
    Inherits ISymbol, IAttributedInstance, IInstance, IBitSize
```

The IValueAnySymbol type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)

	Name	Description
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Methods

	Name	Description
	ReadAnyValue(Type) [▶ 1670]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type
	ReadAnyValue(Type, Int32) [▶ 1670]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type
	UpdateAnyValue(Object.) [▶ 1672]	Reads the value of this Value [▶ 1683] into the specified managed value.
	UpdateAnyValue(Object., Int32) [▶ 1672]	Reads the value of this Value [▶ 1683] into the specified managed value.
	WriteAnyValue(Object) [▶ 1673]	Writes the value represented by the managed value to this Value [▶ 1683]
	WriteAnyValue(Object, Int32) [▶ 1674]	Writes the value represented by the managed value to this Value [▶ 1683]

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.ISymbol [▶ 1634]

5.8.74.1 IValueAnySymbol Properties

The IValueAnySymbol [▶ 1666] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)

	Name	Description
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IValueAnySymbol Interface [▶ 1666]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.74.2 IValueAnySymbol Methods

Methods

	Name	Description
	ReadAnyValue(Type) [▶ 1670]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type
	ReadAnyValue(Type, Int32) [▶ 1670]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type
	UpdateAnyValue(Object.) [▶ 1672]	Reads the value of this Value [▶ 1683] into the specified managed value.
	UpdateAnyValue(Object., Int32) [▶ 1672]	Reads the value of this Value [▶ 1683] into the specified managed value.
	WriteAnyValue(Object) [▶ 1673]	Writes the value represented by the managed value to this Value [▶ 1683]
	WriteAnyValue(Object, Int32) [▶ 1674]	Writes the value represented by the managed value to this Value [▶ 1683]

Reference

IValueAnySymbol Interface [▶ 1666]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.74.2.1 IValueAnySymbol.ReadAnyValue Method

Overload List

	Name	Description
	ReadAnyValue(Type) [▶ 1670]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type
	ReadAnyValue(Type, Int32) [▶ 1670]	Reads the value of this Value [▶ 1683] into a new created instance of the managed type

Reference

IValueAnySymbol Interface [▶ 1666]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueAnySymbol.ReadAnyValue Method (Type)

Reads the value of this Value [▶ 1683] into a new created instance of the managed type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object ReadAnyValue(
    Type managedType
)
```

VB

```
Function ReadAnyValue (
    managedType As Type
) As Object
```

Parameters

managedType	Type: System.Type The tp.

Return Value

Type: Object
Read value (System.Object).

Reference

IValueAnySymbol Interface [▶ 1666]

ReadAnyValue Overload [▶ 1669]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueAnySymbol.WriteAnyValue(Object) [▶ 1673]

IValueAnySymbol.UpdateAnyValue(Object.) [▶ 1672]

IValueAnySymbol.ReadAnyValue Method (Type, Int32)

Reads the value of this Value [▶ 1683] into a new created instance of the managed type

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object ReadAnyValue(
    Type managedType,
    int timeout
)
```

VB

```
Function ReadAnyValue (
    managedType As Type,
    timeout As Integer
) As Object
```

Parameters

managedType	Type: System.Type The tp.
timeout	Type: System.Int32 The timeout in ms.

Return Value

Type: Object
Read value (System.Object).

Reference

IValueAnySymbol Interface [▶ 1666]

ReadAnyValue Overload [▶ 1669]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueAnySymbol.WriteAnyValue(Object) [▶ 1673]

IValueAnySymbol.UpdateAnyValue(Object.) [▶ 1672]

5.8.74.2.2 IValueAnySymbol.UpdateAnyValue Method

Overload List

	Name	Description
	UpdateAnyValue(Object.) [▶ 1672]	Reads the value of this Value [▶ 1683] into the specified managed value.
	UpdateAnyValue(Object., Int32) [▶ 1672]	Reads the value of this Value [▶ 1683] into the specified managed value.

Reference

IValueAnySymbol Interface [▶ 1666]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueAnySymbol.UpdateAnyValue Method (Object.)

Reads the value of this Value [▶ 1683] into the specified managed value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void UpdateAnyValue(
    ref Object managedObject
)
```

VB

```
Sub UpdateAnyValue (
    ByRef managedObject As Object
)
```

Parameters

managedObject	Type: System.Object. The managed object.

Return Value

Type:
Read value (System.Object).

Reference

IValueAnySymbol Interface [▶ 1666]

UpdateAnyValue Overload [▶ 1671]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueAnySymbol.ReadAnyValue(Type) [▶ 1670]

IValueAnySymbol.WriteAnyValue(Object) [▶ 1673]

IValueAnySymbol.UpdateAnyValue Method (Object., Int32)

Reads the value of this Value [▶ 1683] into the specified managed value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void UpdateAnyValue(
    ref Object managedObject,
    int timeout
)
```

VB

```
Sub UpdateAnyValue (
    ByRef managedObject As Object,
    timeout As Integer
)
```

Parameters

managedObject	Type: System.Object. The managed object.
timeout	Type: System.Int32 The timeout.

Return Value

Type:
Read value (System.Object).

Reference

IValueAnySymbol Interface [▶ 1666]

UpdateAnyValue Overload [▶ 1671]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueAnySymbol.ReadAnyValue(Type) [▶ 1670]

IValueAnySymbol.WriteAnyValue(Object) [▶ 1673]

5.8.74.2.3 IValueAnySymbol.WriteAnyValue Method

Overload List

	Name	Description
	WriteAnyValue(Object) [▶ 1673]	Writes the value represented by the managed value to this Value [▶ 1683]
	WriteAnyValue(Object, Int32) [▶ 1674]	Writes the value represented by the managed value to this Value [▶ 1683]

Reference

IValueAnySymbol Interface [▶ 1666]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueAnySymbol.WriteAnyValue Method (Object)

Writes the value represented by the managed value to this Value [▶ 1683]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void WriteAnyValue(
    Object managedValue
)
```

VB

```
Sub WriteAnyValue (
    managedValue As Object
)
```

Parameters

managedValue	Type: System.Object The managed value.

Reference

IValueAnySymbol Interface [▶ 1666]

WriteAnyValue Overload [▶ 1673]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueAnySymbol.ReadAnyValue(Type) [▶ 1670]

IValueAnySymbol.UpdateAnyValue(Object.) [▶ 1672]

IValueAnySymbol.WriteAnyValue Method (Object, Int32)

Writes the value represented by the managed value to this Value [▶ 1683]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void WriteAnyValue(
    Object managedValue,
    int timeout
)
```

VB

```
Sub WriteAnyValue (
    managedValue As Object,
    timeout As Integer
)
```

Parameters

managedValue	Type: System.Object The managed value.
timeout	Type: System.Int32 The timeout in ms.

Reference

IValueAnySymbol Interface [▶ 1666]

WriteAnyValue Overload [▶ 1673]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueAnySymbol.ReadAnyValue(Type) [▶ 1670]

IValueAnySymbol.UpdateAnyValue(Object.) [▶ 1672]

5.8.75 IValueRawSymbol Interface

Interface IValueRawSymbol

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IValueRawSymbol : ISymbol,
    IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IValueRawSymbol
    Inherits ISymbol, IAttributedInstance, IInstance, IBitSize
```

The IValueRawSymbol type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasValue [▶ 1678]	Gets a value indicating whether this IValueSymbol [▶ 1683] has a value.
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)

	Name	Description
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Methods

	Name	Description
	ReadRawValue. [▶ 1679]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	ReadRawValue(Int32) [▶ 1680]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	WriteRawValue(.Byte.) [▶ 1681]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	WriteRawValue(.Byte., Int32) [▶ 1682]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Events

	Name	Description
	RawValueChanged [▶ 1683]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.75.1 IValueRawSymbol Properties

The IValueRawSymbol [▶ 1675] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasValue [▶ 1678]	Gets a value indicating whether this IValueSymbol [▶ 1683] has a value.
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)

	Name	Description
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IValueRawSymbol Interface [▶ 1675]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.75.1.1 IValueRawSymbol.HasValue Property

Gets a value indicating whether this IValueSymbol [▶ 1683] has a value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool HasValue { get; }
```

VB

```
ReadOnly Property HasValue As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance has value; otherwise, false.

Remarks

A VirtualSymbol does not support values, but in terms of the IValueSymbol [▶ 1683] definition, is a IValueSymbol [▶ 1683]

Reference

IValueRawSymbol Interface [▶ 1675]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.75.2 IValueRawSymbol Methods

Methods

	Name	Description
	ReadRawValue. [▶ 1679]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	ReadRawValue(Int32) [▶ 1680]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

	Name	Description
	WriteRawValue(.Byte.) [▶ 1681]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	WriteRawValue(.Byte., Int32) [▶ 1682]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Reference

IValueRawSymbol Interface [▶ 1675]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.75.2.1 IValueRawSymbol.ReadRawValue Method

Overload List

	Name	Description
	ReadRawValue. [▶ 1679]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	ReadRawValue(Int32) [▶ 1680]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Reference

IValueRawSymbol Interface [▶ 1675]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueRawSymbol.ReadRawValue Method

Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
byte[] ReadRawValue()
```

VB

```
Function ReadRawValue As Byte()
```

Field Value

Type: .Byte.
The raw value.

Return Value

Type: .Byte.
System.Byte[].

Reference

IValueRawSymbol Interface [▶ 1675]

ReadRawValue Overload [▶ 1679]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueRawSymbol.ReadRawValue Method (Int32)

Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
byte[] ReadRawValue(
    int timeout
)
```

VB

```
Function ReadRawValue (
    timeout As Integer
) As Byte()
```

Parameters

timeout	Type: System.Int32 The timeout in ms.

Field Value

Type: .Byte.
The raw value.

Return Value

Type: .Byte.
System.Byte[].

Remarks

A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

IValueRawSymbol Interface [▶ 1675]

ReadRawValue Overload [▶ 1679]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.75.2.2 IValueRawSymbol.WriteRawValue Method

Overload List

	Name	Description
	WriteRawValue(.Byte.) [▶ 1681]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)
	WriteRawValue(.Byte., Int32) [▶ 1682]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Reference

IValueRawSymbol Interface [▶ 1675]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueRawSymbol.WriteRawValue Method (.Byte.)

Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void WriteRawValue(
    byte[] value
)
```

VB

```
Sub WriteRawValue (
    value As Byte()
)
```

Parameters

value	Type: .System.Byte. The value as byte array.

Field Value

Type:
The value.

Reference

IValueRawSymbol Interface [▶ 1675]

WriteRawValue Overload [▶ 1681]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueRawSymbol.WriteRawValue Method (.Byte., Int32)

Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void WriteRawValue(
    byte[] value,
    int timeout
)
```

VB

```
Sub WriteRawValue (
    value As Byte(),
    timeout As Integer
)
```

Parameters

value	Type: .System.Byte. The value as byte array.
timeout	Type: System.Int32 The timeout.

Field Value

Type:
The value.

Remarks

A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

IValueRawSymbol Interface [▶ 1675]

WriteRawValue Overload [▶ 1681]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.75.3 IValueRawSymbol Events

The IValueRawSymbol [▶ 1675] type exposes the following members.

Events

	Name	Description
	RawValueChanged [▶ 1683]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed.

Reference

IValueRawSymbol Interface [▶ 1675]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.75.3.1 IValueRawSymbol.RawValueChanged Event

Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
event EventHandler<RawValueChangedArgs> RawValueChanged
```

VB

```
Event RawValueChanged As EventHandler(Of RawValueChangedArgs)
```

Value

Type: System.EventHandler.RawValueChangedArgs [▶ 1725].

Reference

IValueRawSymbol Interface [▶ 1675]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.76 IValueSymbol Interface

Interface for a ISymbol [▶ 1634] that supports values.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IValueSymbol : IValueRawSymbol,
    ISymbol, IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IValueSymbol
    Inherits IValueRawSymbol, ISymbol, IAttributedInstance, IInstance,
    IBitSize
```

The IValueSymbol type exposes the following members.

Properties

	Name	Description
	AccessRights [▶ 1687]	Gets the access rights.
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)

	Name	Description
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasValue [▶ 1678]	Gets a value indicating whether this IValueSymbol has a value. (Inherited from IValueRawSymbol [▶ 1675].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	NotificationSettings [▶ 1688]	Gets or sets the notification settings.
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Methods

	Name	Description
	ReadRawValue. [▶ 1679]	Reads the raw value of the IValueSymbol (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	ReadRawValue(Int32) [▶ 1680]	Reads the raw value of the IValueSymbol (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	ReadValue. [▶ 1690]	Reads the Value of the IValueSymbol
	ReadValue(Int32) [▶ 1691]	Reads the Value of the IValueSymbol
	WriteRawValue(.Byte.) [▶ 1681]	Writes the raw value of the IValueSymbol (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	WriteRawValue(.Byte., Int32) [▶ 1682]	Writes the raw value of the IValueSymbol (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	WriteValue(Object) [▶ 1692]	Writes the Value of the IValueSymbol
	WriteValue(Object, Int32) [▶ 1693]	Writes the Value of the IValueSymbol

Events

	Name	Description
	RawValueChanged [▶ 1683]	Occurs when the RawValue of the IValueSymbol has changed. (Inherited from IValueRawSymbol [▶ 1675].)
	ValueChanged [▶ 1694]	Occurs when the (Primitive) value of the IValueSymbol has changed.

Extension Methods

	Name	Description
	PollValuesAnnotated(IObservable.Unit.) [▶ 882]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence (Defined by ValueSymbolExtensions [▶ 877].)
	PollValuesAnnotated(TimeSpan) [▶ 883]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time. (Defined by ValueSymbolExtensions [▶ 877].)
	WhenValueChanged [▶ 884]	Gets an observable sequence when the value of the IValueSymbol has changed. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object.) [▶ 888]	Overloaded. Subscribes the IValueSymbol to an observable sequence of values and writes them to the IValueSymbol. (Defined by ValueSymbolExtensions [▶ 877].)

	Name	Description
	WriteValues(IObservable.Object., Action.Exception.) [▶ 890]	Overloaded. Subscribes the IValueSymbol to an observable sequence of values and writes them to the IValueSymbol. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., CancellationToken) [▶ 890]	Overloaded. Subscribes the IValueSymbol to an observable sequence of values and writes them to the IValueSymbol. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception., CancellationToken) [▶ 891]	Overloaded. Subscribes the IValueSymbol to an observable sequence of values and writes them to the IValueSymbol. (Defined by ValueSymbolExtensions [▶ 877].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IValueRawSymbol [▶ 1675]

5.8.76.1 IValueSymbol Properties

The IValueSymbol [▶ 1683] type exposes the following members.

Properties

	Name	Description
	AccessRights [▶ 1687]	Gets the access rights.
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasValue [▶ 1678]	Gets a value indicating whether this IValueSymbol [▶ 1683] has a value. (Inherited from IValueRawSymbol [▶ 1675].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)

	Name	Description
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	NotificationSettings [▶ 1688]	Gets or sets the notification settings.
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IValueSymbol Interface [▶ 1683]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.76.1.1 IValueSymbol.AccessRights Property

Gets the access rights.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
SymbolAccessRights AccessRights { get; }
```

VB

```
ReadOnly Property AccessRights As SymbolAccessRights
    Get
```

Property Value

Type: SymbolAccessRights [▶ 1818]
The access rights.

Reference

IValueSymbol Interface [▶ 1683]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.76.1.2 IValueSymbol.NotificationSettings Property

Gets or sets the notification settings.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
INotificationSettings NotificationSettings { get; set; }
```

VB

```
Property NotificationSettings As INotificationSettings
    Get
    Set
```

Property Value

Type: INotificationSettings [▶ 1566]
The notification settings.

Remarks

The NotificationSettings will be inherited from Parent [▶ 1639] if the setting is not overwritten.

Reference

IValueSymbol Interface [▶ 1683]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.76.2 IValueSymbol Methods

The IValueSymbol [▶ 1683] type exposes the following members.

Methods

	Name	Description
	ReadRawValue. [▶ 1679]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)

	Name	Description
	ReadRawValue(Int32) [▶ 1680]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	ReadValue. [▶ 1690]	Reads the Value of the IValueSymbol [▶ 1683]
	ReadValue(Int32) [▶ 1691]	Reads the Value of the IValueSymbol [▶ 1683]
	WriteRawValue(.Byte.) [▶ 1681]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	WriteRawValue(.Byte., Int32) [▶ 1682]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	WriteValue(Object) [▶ 1692]	Writes the Value of the IValueSymbol [▶ 1683]
	WriteValue(Object, Int32) [▶ 1693]	Writes the Value of the IValueSymbol [▶ 1683]

Extension Methods

	Name	Description
	PollValuesAnnotated(IObservable.Unit.) [▶ 882]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence (Defined by ValueSymbolExtensions [▶ 877].)
	PollValuesAnnotated(TimeSpan) [▶ 883]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time. (Defined by ValueSymbolExtensions [▶ 877].)
	WhenValueChanged [▶ 884]	Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object.) [▶ 888]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception.) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., CancellationToken) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception., CancellationToken) [▶ 891]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)

Reference

IValueSymbol Interface [▶ 1683]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.76.2.1 IValueSymbol.ReadValue Method

Overload List

	Name	Description
	ReadValue. [▶ 1690]	Reads the Value of the IValueSymbol [▶ 1683]
	ReadValue(Int32) [▶ 1691]	Reads the Value of the IValueSymbol [▶ 1683]

Reference

IValueSymbol Interface [▶ 1683]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueSymbol.ReadValue Method

Reads the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object ReadValue()
```

VB

```
Function ReadValue As Object
```

Field Value

Type: Object
The value.

Return Value

Type: Object
System.Object.

Remarks

Calling on primitive types, a call of this method will return the primitive value. On complex types (structures and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: the raw byte Array will be returned, in dynamic mode: A Value will be created on the fly.

Reference

IValueSymbol Interface [▶ 1683]

ReadValue Overload [▶ 1690]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueSymbol.ReadValue Method (Int32)

Reads the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
Object ReadValue(
    int timeout
)
```

VB

```
Function ReadValue (
    timeout As Integer
) As Object
```

Parameters

timeout	Type: System.Int32 The timeout in ms.

Field Value

Type: Object
The value.

Return Value

Type: Object
System.Object.

Remarks

Calling on primitive types, a call of this method will return the primitive value. On complex types (structures and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: the raw byte Array will be returned, in dynamic mode: A Value will be created on the fly. A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

IValueSymbol Interface [▶ 1683]

ReadValue Overload [▶ 1690]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.76.2.2 IValueSymbol.WriteValue Method

Overload List

	Name	Description
	WriteValue(Object) [▶ 1692]	Writes the Value of the IValueSymbol [▶ 1683]
	WriteValue(Object, Int32) [▶ 1693]	Writes the Value of the IValueSymbol [▶ 1683]

Reference

IValueSymbol Interface [▶ 1683]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueSymbol.WriteValue Method (Object)

Writes the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void WriteValue(
    Object value
)
```

VB

```
Sub WriteValue (
    value As Object
)
```

Parameters

value	Type: System.Object The value.

Remarks

Calling on primitive types, a call of this method will directly write this Value. On complex types (structs and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: Only byte Arrays (of correct size) can be written) in dynamic mode: A Value that represents the value will be accepted also.

Reference

IValueSymbol Interface [▶ 1683]

WriteValue Overload [▶ 1692]

TwinCAT.TypeSystem Namespace [▶ 1184]

IValueSymbol.WriteValue Method (Object, Int32)

Writes the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
void WriteValue(
    Object value,
    int timeout
)
```

VB

```
Sub WriteValue (
    value As Object,
    timeout As Integer
)
```

Parameters

value	Type: System.Object The value.
timeout	Type: System.Int32 The timeout in ms.

Remarks

Calling on primitive types, a call of this method will directly write this Value. On complex types (structs and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: Only byte Arrays (of correct size) can be written) in dynamic mode: A Value that represents the value will be accepted also. A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

IValueSymbol Interface [▶ 1683]

WriteValue Overload [▶ 1692]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.76.3 IValueSymbol Events

The IValueSymbol [▶ 1683] type exposes the following members.

Events

	Name	Description
	RawValueChanged [▶ 1683]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from IValueRawSymbol [▶ 1675].)
	ValueChanged [▶ 1694]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed.

Reference

IValueSymbol Interface [▶ 1683]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.76.3.1 IValueSymbol.ValueChanged Event

Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
event EventHandler<ValueChangedArgs> ValueChanged
```

VB

```
Event ValueChanged As EventHandler(Of ValueChangedArgs)
```

Value

Type: System.EventHandler.ValueChangedArgs [▶ 1847].

Reference

IValueSymbol Interface [▶ 1683]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.77 IValueSymbol2 Interface

Interface for a ISymbol [▶ 1634] that supports values.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IValueSymbol2 : IValueSymbol,
    IValueRawSymbol, ISymbol, IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IValueSymbol2
    Inherits IValueSymbol, IValueRawSymbol, ISymbol, IAttributedInstance,
    IInstance, IBitSize
```

The IValueSymbol2 type exposes the following members.

Properties

	Name	Description
	AccessRights [▶ 1687]	Gets the access rights. (Inherited from IValueSymbol [▶ 1683].)

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	Connection [▶ 1698]	Gets the connection that produces values for this IValueSymbol [▶ 1683]
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasValue [▶ 1678]	Gets a value indicating whether this IValueSymbol [▶ 1683] has a value. (Inherited from IValueRawSymbol [▶ 1675].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	NotificationSettings [▶ 1688]	Gets or sets the notification settings. (Inherited from IValueSymbol [▶ 1683].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)

	Name	Description
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Methods

	Name	Description
	ReadRawValue. [▶ 1679]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	ReadRawValue(Int32) [▶ 1680]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	ReadValue. [▶ 1690]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)
	ReadValue(Int32) [▶ 1691]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)
	WriteRawValue(.Byte.) [▶ 1681]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	WriteRawValue(.Byte., Int32) [▶ 1682]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	WriteValue(Object) [▶ 1692]	Writes the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)
	WriteValue(Object, Int32) [▶ 1693]	Writes the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)

Events

	Name	Description
	RawValueChanged [▶ 1683]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from IValueRawSymbol [▶ 1675].)
	ValueChanged [▶ 1694]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from IValueSymbol [▶ 1683].)

Extension Methods

	Name	Description
	PollValuesAnnotated(IObservable.Unit.) [▶ 882]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence (Defined by ValueSymbolExtensions [▶ 877].)
	PollValuesAnnotated(TimeSpan) [▶ 883]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time. (Defined by ValueSymbolExtensions [▶ 877].)
	WhenValueChanged [▶ 884]	Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object.) [▶ 888]	Overloaded.

	Name	Description
		Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception.) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., CancellationToken) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception., CancellationToken) [▶ 891]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IValueSymbol [▶ 1683]

5.8.77.1 IValueSymbol2 Properties

The IValueSymbol2 [▶ 1694] type exposes the following members.

Properties

	Name	Description
	AccessRights [▶ 1687]	Gets the access rights. (Inherited from IValueSymbol [▶ 1683].)
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	Connection [▶ 1698]	Gets the connection that produces values for this IValueSymbol [▶ 1683]
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasValue [▶ 1678]	Gets a value indicating whether this IValueSymbol [▶ 1683] has a value. (Inherited from IValueRawSymbol [▶ 1675].)

	Name	Description
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	NotificationSettings [▶ 1688]	Gets or sets the notification settings. (Inherited from IValueSymbol [▶ 1683].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IValueSymbol2 Interface [▶ 1694]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.77.1.1 IValueSymbol2.Connection Property

Gets the connection that produces values for this IValueSymbol [▶ 1683]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IConnection Connection { get; }
```

VB

```
ReadOnly Property Connection As IConnection
    Get
```

Property Value

Type: IConnection [▶ 46]
The connection object.

Reference

IValueSymbol2 Interface [▶ 1694]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.77.2 IValueSymbol2 Methods

The IValueSymbol2 [▶ 1694] type exposes the following members.

Methods

	Name	Description
	ReadRawValue. [▶ 1679]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	ReadRawValue(Int32) [▶ 1680]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	ReadValue. [▶ 1690]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)
	ReadValue(Int32) [▶ 1691]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)
	WriteRawValue(.Byte.) [▶ 1681]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	WriteRawValue(.Byte., Int32) [▶ 1682]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	WriteValue(Object) [▶ 1692]	Writes the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)
	WriteValue(Object, Int32) [▶ 1693]	Writes the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)

Extension Methods

	Name	Description
	PollValuesAnnotated(IObservable.Unit.) [▶ 882]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence (Defined by ValueSymbolExtensions [▶ 877].)

	Name	Description
	PollValuesAnnotated(TimeSpan) [▶ 883]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time. (Defined by ValueSymbolExtensions [▶ 877].)
	WhenValueChanged [▶ 884]	Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object.) [▶ 888]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception.) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., CancellationToken) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception., CancellationToken) [▶ 891]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)

Reference

IValueSymbol2 Interface [▶ 1694]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.77.3 IValueSymbol2 Events

The IValueSymbol2 [▶ 1694] type exposes the following members.

Events

	Name	Description
	RawValueChanged [▶ 1683]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from IValueRawSymbol [▶ 1675].)
	ValueChanged [▶ 1694]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from IValueSymbol [▶ 1683].)

Reference

IValueSymbol2 Interface [▶ 1694]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.78 IValueSymbol3 Interface

Interface IValueSymbol3 Implements the IValueSymbol2 [▶ 1694]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IValueSymbol3 : IValueSymbol2,
    IValueSymbol, IValueRawSymbol, ISymbol, IAttributedInstance, IInstance,
    IBitSize
```

VB

```
Public Interface IValueSymbol3
    Inherits IValueSymbol2, IValueSymbol, IValueRawSymbol, ISymbol,
    IAttributedInstance, IInstance, IBitSize
```

The IValueSymbol3 type exposes the following members.

Properties

	Name	Description
	AccessRights [▶ 1687]	Gets the access rights. (Inherited from IValueSymbol [▶ 1683].)
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	Connection [▶ 1698]	Gets the connection that produces values for this IValueSymbol [▶ 1683] (Inherited from IValueSymbol2 [▶ 1694].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasValue [▶ 1678]	Gets a value indicating whether this IValueSymbol [▶ 1683] has a value. (Inherited from IValueRawSymbol [▶ 1675].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)

	Name	Description
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	NotificationSettings [▶ 1688]	Gets or sets the notification settings. (Inherited from IValueSymbol [▶ 1683].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Methods

	Name	Description
	ReadRawValue. [▶ 1679]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	ReadRawValue(Int32) [▶ 1680]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	ReadValue. [▶ 1690]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)
	ReadValue(Int32) [▶ 1691]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)
	TryReadValue [▶ 1706]	Reads the Value of the IValueSymbol [▶ 1683]
	TryWriteValue [▶ 1707]	Writes the Value of the IValueSymbol [▶ 1683]
	WriteRawValue(.Byte.) [▶ 1681]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	WriteRawValue(.Byte., Int32) [▶ 1682]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	WriteValue(Object) [▶ 1692]	Writes the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)

	Name	Description
	WriteValue(Object, Int32) [▶ 1693]	Writes the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)

Events

	Name	Description
	RawValueChanged [▶ 1683]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from IValueRawSymbol [▶ 1675].)
	ValueChanged [▶ 1694]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from IValueSymbol [▶ 1683].)

Extension Methods

	Name	Description
	PollValuesAnnotated(IObservable.Unit.) [▶ 882]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence (Defined by ValueSymbolExtensions [▶ 877].)
	PollValuesAnnotated(TimeSpan) [▶ 883]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time. (Defined by ValueSymbolExtensions [▶ 877].)
	WhenValueChanged [▶ 884]	Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object.) [▶ 888]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception.) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., CancellationToken) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception., CancellationToken) [▶ 891]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IValueSymbol2 [▶ 1694]

5.8.78.1 IValueSymbol3 Properties

The IValueSymbol3 [▶ 1701] type exposes the following members.

Properties

	Name	Description
	AccessRights [▶ 1687]	Gets the access rights. (Inherited from IValueSymbol [▶ 1683].)
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	Connection [▶ 1698]	Gets the connection that produces values for this IValueSymbol [▶ 1683] (Inherited from IValueSymbol2 [▶ 1694].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasValue [▶ 1678]	Gets a value indicating whether this IValueSymbol [▶ 1683] has a value. (Inherited from IValueRawSymbol [▶ 1675].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)

	Name	Description
	NotificationSettings [▶ 1688]	Gets or sets the notification settings. (Inherited from IValueSymbol [▶ 1683].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IValueSymbol3 Interface [▶ 1701]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.78.2 IValueSymbol3 Methods

The IValueSymbol3 [▶ 1701] type exposes the following members.

Methods

	Name	Description
	ReadRawValue. [▶ 1679]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	ReadRawValue(Int32) [▶ 1680]	Reads the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	ReadValue. [▶ 1690]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)
	ReadValue(Int32) [▶ 1691]	Reads the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)
	TryReadValue [▶ 1706]	Reads the Value of the IValueSymbol [▶ 1683]
	TryWriteValue [▶ 1707]	Writes the Value of the IValueSymbol [▶ 1683]
	WriteRawValue(.Byte.) [▶ 1681]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	WriteRawValue(.Byte., Int32) [▶ 1682]	Writes the raw value of the IValueSymbol [▶ 1683] (Ads Read / Write) (Inherited from IValueRawSymbol [▶ 1675].)
	WriteValue(Object) [▶ 1692]	Writes the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)
	WriteValue(Object, Int32) [▶ 1693]	Writes the Value of the IValueSymbol [▶ 1683] (Inherited from IValueSymbol [▶ 1683].)

Extension Methods

	Name	Description
	PollValuesAnnotated(IObservable.Unit.) [▶ 882]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence annotated value on trigger sequence (Defined by ValueSymbolExtensions [▶ 877].)
	PollValuesAnnotated(TimeSpan) [▶ 883]	Overloaded. Polls the values as ValueChangedArgs [▶ 1847] sequence with a specified period time. (Defined by ValueSymbolExtensions [▶ 877].)
	WhenValueChanged [▶ 884]	Gets an observable sequence when the value of the IValueSymbol [▶ 1683] has changed. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object.) [▶ 888]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception.) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., CancellationToken) [▶ 890]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)
	WriteValues(IObservable.Object., Action.Exception., CancellationToken) [▶ 891]	Overloaded. Subscribes the IValueSymbol [▶ 1683] to an observable sequence of values and writes them to the IValueSymbol [▶ 1683]. (Defined by ValueSymbolExtensions [▶ 877].)

Reference

IValueSymbol3 Interface [▶ 1701]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.78.2.1 IValueSymbol3.TryReadValue Method

Reads the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int TryReadValue(
    int timeout,
    out Object value
)
```

VB

```
Function TryReadValue (
    timeout As Integer,
    <OutAttribute> ByRef value As Object
) As Integer
```

Parameters

timeout	Type: System.Int32 The timeout in ms.
value	Type: System.Object. The symbol value.

Return Value

Type: Int32
The error code.

Remarks

Calling on primitive types, a call of this method will return the primitive value. On complex types (structures and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: the raw byte Array will be returned, in dynamic mode: A Value will be created on the fly. A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

IValueSymbol3 Interface [▶ 1701]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.78.2.2 IValueSymbol3.TryWriteValue Method

Writes the Value of the IValueSymbol [▶ 1683]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
int TryWriteValue(
    Object value,
    int timeout
)
```

VB

```
Function TryWriteValue (
    value As Object,
    timeout As Integer
) As Integer
```

Parameters

value	Type: System.Object The value.
timeout	Type: System.Int32 The timeout in ms.

Return Value

Type: Int32
The error code.

Remarks

Calling on primitive types, a call of this method will directly write this Value. On complex types (structs and arrays) it depends on the ISymbolLoader [▶ 1645] settings what will happen. In non dynamic modes: Only byte Arrays (of correct size) can be written) in dynamic mode: A Value that represents the value will be accepted also. A negative timeout indicates that the Default Timeout for the communication will be used. 0 means timeout is switched off.

Reference

IValueSymbol3 Interface [▶ 1701]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.78.3 IValueSymbol3 Events

The IValueSymbol3 [▶ 1701] type exposes the following members.

Events

	Name	Description
⚡	RawValueChanged [▶ 1683]	Occurs when the RawValue of the IValueSymbol [▶ 1683] has changed. (Inherited from IValueRawSymbol [▶ 1675].)
⚡	ValueChanged [▶ 1694]	Occurs when the (Primitive) value of the IValueSymbol [▶ 1683] has changed. (Inherited from IValueSymbol [▶ 1683].)

Reference

IValueSymbol3 Interface [▶ 1701]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.79 IVirtualStructInstance Interface

Virtual Struct instance interface.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IVirtualStructInstance : IStructInstance,
    ISymbol, IAttributedInstance, IInstance, IBitSize
```

VB

```
Public Interface IVirtualStructInstance
    Inherits IStructInstance, ISymbol, IAttributedInstance, IInstance,
    IBitSize
```

The IVirtualStructInstance type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasRpcMethods [▶ 1618]	Gets a value indicating whether this instance has RPC methods (Inherited from IStructInstance [▶ 1615].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	MemberInstances [▶ 1619]	Gets the member instances of the Struct Instance [▶ 1615]. (Inherited from IStructInstance [▶ 1615].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)

	Name	Description
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Methods

	Name	Description
	AddMember [▶ 1711]	Adds the member.

Remarks

Virtual struct instance are used to create a TreeView from the flat list of symbols.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

TwinCAT.TypeSystem.IStructInstance [▶ 1615]

5.8.79.1 IVirtualStructInstance Properties

The IVirtualStructInstance [▶ 1708] type exposes the following members.

Properties

	Name	Description
	Attributes [▶ 1513]	Gets the Type Attributes. (Inherited from IAttributedInstance [▶ 1512].)
	BitSize [▶ 1515]	Gets the size of the IDataType [▶ 1517] in bits. (Inherited from IBitSize [▶ 1514].)
	ByteSize [▶ 1516]	Gets the (aligned) size of of the Type/Instance in Bytes (Inherited from IBitSize [▶ 1514].)
	Category [▶ 1636]	Gets the Symbol/Datatype Category (Inherited from ISymbol [▶ 1634].)
	Comment [▶ 1558]	Gets the comment of the IInstance [▶ 1556] (Inherited from IInstance [▶ 1556].)
	DataType [▶ 1559]	Gets the IDataType [▶ 1517] of the IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)
	HasRpcMethods [▶ 1618]	Gets a value indicating whether this instance has RPC methods (Inherited from IStructInstance [▶ 1615].)
	InstanceName [▶ 1559]	Gets the name of the instance (without periods (.) (Inherited from IInstance [▶ 1556].)
	InstancePath [▶ 1560]	Gets the relative / absolute access path to the instance (with periods (.)) (Inherited from IInstance [▶ 1556].)
	IsBitType [▶ 1516]	Gets a value indicating whether this instance is not basing on a full DataType but instead of some sort of bit mapping (Inherited from IBitSize [▶ 1514].)
	IsByteAligned [▶ 1517]	Indicates that the Size of the Object is Byte aligned (BitSize % 8 == 0) (Inherited from IBitSize [▶ 1514].)
	IsContainerType [▶ 1637]	Gets a value indicating whether this Symbol is a container type. (Inherited from ISymbol [▶ 1634].)

	Name	Description
	IsPersistent [▶ 1637]	Gets a value indicating whether this ISymbol [▶ 1634] is persistent. (Inherited from ISymbol [▶ 1634].)
	IsPointer [▶ 1560]	Indicates that the IInstance [▶ 1556] represents a Pointer type (Pointer TO) (Inherited from IInstance [▶ 1556].)
	IsPrimitiveType [▶ 1638]	Gets a value indicating whether this instance is a primitive type. (Inherited from ISymbol [▶ 1634].)
	IsReadOnly [▶ 1638]	Gets a value indicating whether this ISymbol [▶ 1634] is read only. (Inherited from ISymbol [▶ 1634].)
	IsRecursive [▶ 1639]	Gets a value indicating whether this instance is recursive. (Inherited from ISymbol [▶ 1634].)
	IsReference [▶ 1561]	Indicates that the IInstance [▶ 1556] represents a Reference type (REFERENCE TO) (Inherited from IInstance [▶ 1556].)
	IsStatic [▶ 1561]	Gets a value indicating whether this IInstance [▶ 1556] is static. (Inherited from IInstance [▶ 1556].)
	MemberInstances [▶ 1619]	Gets the member instances of the Struct Instance [▶ 1615]. (Inherited from IStructInstance [▶ 1615].)
	Parent [▶ 1639]	Gets the parent Symbol (Inherited from ISymbol [▶ 1634].)
	Size [▶ 1517]	Gets the size of the object in bytes or Bits dependant on IsBitType [▶ 1516] (Inherited from IBitSize [▶ 1514].)
	SubSymbols [▶ 1640]	Gets the SubSymbols of the ISymbol [▶ 1634] (Inherited from ISymbol [▶ 1634].)
	TypeName [▶ 1562]	Gets the name of the DataType [▶ 1517] that is used for this IInstance [▶ 1556]. (Inherited from IInstance [▶ 1556].)

Reference

IVirtualStructInstance Interface [▶ 1708]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.79.2 IVirtualStructInstance Methods

The IVirtualStructInstance [▶ 1708] type exposes the following members.

Methods

	Name	Description
	AddMember [▶ 1711]	Adds the member.

Reference

IVirtualStructInstance Interface [▶ 1708]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.79.2.1 IVirtualStructInstance.AddMember Method

Adds the member.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool AddMember(
    ISymbol memberInstance,
    IVirtualStructInstance parent
)
```

VB

```
Function AddMember (
    memberInstance As ISymbol,
    parent As IVirtualStructInstance
) As Boolean
```

Parameters

memberInstance	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The member instance.
parent	Type: TwinCAT.TypeSystem.IVirtualStructInstance [▶ 1708] The parent struct instance. Usually the this pointer.

Return Value

Type: Boolean

Reference

IVirtualStructInstance Interface [▶ 1708]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.80 MarshalException Class

Common Marshalling Exception

Inheritance Hierarchy

System.Object
  System.Exception
    System.ApplicationException
      TwinCAT.Ads.AdsException [▶ 318]
        TwinCAT.TypeSystem.MarshalException

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[SerializableAttribute]
public class MarshalException : AdsException
```

VB

```
<SerializableAttribute>
Public Class MarshalException
    Inherits AdsException
```

The MarshalException type exposes the following members.

Constructors

	Name	Description
	MarshalException. [▶ 1714]	Initializes a new instance of the MarshalException class.
	MarshalException(String) [▶ 1715]	Initializes a new instance of the MarshalException class.
	MarshalException(String, Exception) [▶ 1715]	Initializes a new Instance of the AdsException class.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)

	Name	Description
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.80.1 MarshalException Constructor

Overload List

	Name	Description
	MarshalException. [▶ 1714]	Initializes a new instance of the MarshalException [▶ 1712] class.
	MarshalException(String) [▶ 1715]	Initializes a new instance of the MarshalException [▶ 1712] class.
	MarshalException(String, Exception) [▶ 1715]	Initializes a new Instance of the AdsException class.

Reference

MarshalException Class [▶ 1712]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.80.1.1 MarshalException Constructor

Initializes a new instance of the MarshalException [▶ 1712] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public MarshalException()
```

VB

```
Public Sub New
```

Reference

MarshalException Class [▶ 1712]

MarshalException Overload [▶ 1714]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.80.1.2 MarshalException Constructor (String)

Initializes a new instance of the MarshalException [▶ 1712] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public MarshalException(
    string message
)
```

VB

```
Public Sub New (
    message As String
)
```

Parameters

message	Type: System.String The message.

Reference

MarshalException Class [▶ 1712]

MarshalException Overload [▶ 1714]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.80.1.3 MarshalException Constructor (String, Exception)

Initializes a new Instance of the AdsException class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public MarshalException(
    string message,
    Exception innerException
)
```

VB

```
Public Sub New (
    message As String,
    innerException As Exception
)
```

Parameters

message	Type: System.String The message.
innerException	Type: System.Exception The inner exception.

Reference

MarshalException Class [▶ 1712]

MarshalException Overload [▶ 1714]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.80.2 MarshalException Properties

The MarshalException [▶ 1712] type exposes the following members.

Properties

	Name	Description
	Data	Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)
	HelpLink	Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)
	HResult	Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)
	InnerException	Gets the Exception instance that caused the current exception. (Inherited from Exception.)
	Message	Gets a message that describes the current exception. (Inherited from Exception.)
	Source	Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)
	StackTrace	Gets a string representation of the immediate frames on the call stack. (Inherited from Exception.)
	TargetSite	Gets the method that throws the current exception. (Inherited from Exception.)

Reference

MarshalException Class [▶ 1712]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.80.3 MarshalException Methods

The MarshalException [▶ 1712] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)

	Name	Description
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetBaseException	When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetObjectData	When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)
	GetType	Gets the runtime type of the current instance. (Inherited from Exception.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Creates and returns a string representation of the current exception. (Inherited from Exception.)

Reference

MarshalException Class [▶ 1712]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.80.4 MarshalException Events

The MarshalException [▶ 1712] type exposes the following members.

Events

	Name	Description
	SerializeObjectState	Occurs when an exception is serialized to create an exception state object that contains serialized data about the exception. (Inherited from Exception.)

Reference

MarshalException Class [▶ 1712]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.81 MemberCollection Class

Collection of IMember [▶ 1562] objects.

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.Generic.InstanceCollection [▶ 1887].IMember [▶ 1562].
    TwinCAT.TypeSystem.MemberCollection

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class MemberCollection : InstanceCollection<IMember>
```

VB

```
Public Class MemberCollection
    Inherits InstanceCollection(Of IMember)
```

The MemberCollection type exposes the following members.

Constructors

	Name	Description
	MemberCollection. [▶ 1720]	Initializes a new instance of the MemberCollection class.
	MemberCollection(IEnumerable.IMember.) [▶ 1720]	Initializes a new instance of the MemberCollection class (copy constructor)

Properties

	Name	Description
	Count [▶ 1891]	Gets the collection count. (Inherited from InstanceCollection.T. [▶ 1887].)
	IsReadOnly [▶ 1892]	Gets a value indicating whether this instance is read only. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.Int32. [▶ 1893]	Gets or sets the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.String. [▶ 1894]	Gets the IInstance [▶ 1556] with the specified instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	Mode [▶ 1894]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from InstanceCollection.T. [▶ 1887].)

Methods

	Name	Description
	Add [▶ 1896]	Adds the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	AddRange [▶ 1897]	Adds the specified items to this collection. (Inherited from InstanceCollection.T. [▶ 1887].)
	AsReadOnly [▶ 1722]	Returns a read only copy of this collection (shallow copy)
	Clear [▶ 1898]	Clears this instance. (Inherited from InstanceCollection.T. [▶ 1887].)
	Clone [▶ 1723]	Clones this MemberCollection.
	Contains(String) [▶ 1899]	Determines whether this collection contains an IInstance [▶ 1556] with the specified InstanceName / InstancePath (Inherited from InstanceCollection.T. [▶ 1887].)
	Contains(T) [▶ 1899]	Determines whether this collection contains the specified IInstance [▶ 1556] (Inherited from InstanceCollection.T. [▶ 1887].)
	ContainsName [▶ 1900]	Determines whether the specified instance name contains name. (Inherited from InstanceCollection.T. [▶ 1887].)

	Name	Description
	CopyTo [▶ 1901]	Copies this InstanceCollection.T. [▶ 1887] to the specified array. (Inherited from InstanceCollection.T. [▶ 1887].)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1902]	Gets the enumerator. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1902]	Gets the IInstance [▶ 1556]by instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetInstanceByName [▶ 1903]	Gets the name of the instance by. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1904]	Determines the index of the specified IInstance [▶ 1556]. (Inherited from InstanceCollection.T. [▶ 1887].)
	Insert [▶ 1905]	Inserts the specified IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1905]	Removes the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	RemoveAt [▶ 1906]	Removes the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1907]	Tries to get the IInstance [▶ 1556]. of the specified path. (Inherited from InstanceCollection.T. [▶ 1887].)
	TryGetInstanceByName [▶ 1908]	Tries to get Instnaces by name. (Inherited from InstanceCollection.T. [▶ 1887].)

Fields

	Name	Description
	_list [▶ 1909]	The _list (Inherited from InstanceCollection.T. [▶ 1887].)
	_pathDict [▶ 1909]	The _path dictionary (Inherited from InstanceCollection.T. [▶ 1887].)
	mode [▶ 1910]	The mode this InstanceCollection.T. [▶ 1887] is working in. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.81.1 MemberCollection Constructor

Overload List

	Name	Description
	MemberCollection. [▶ 1720]	Initializes a new instance of the MemberCollection [▶ 1717] class.
	MemberCollection(IEnumerable.IMember.) [▶ 1720]	Initializes a new instance of the MemberCollection [▶ 1717] class (copy constructor)

Reference

MemberCollection Class [▶ 1717]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.81.1.1 MemberCollection Constructor

Initializes a new instance of the MemberCollection [▶ 1717] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public MemberCollection()
```

VB

```
Public Sub New
```

Reference

MemberCollection Class [▶ 1717]

MemberCollection Overload [▶ 1720]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.81.1.2 MemberCollection Constructor (IEnumerable.IMember.)

Initializes a new instance of the MemberCollection [▶ 1717] class (copy constructor)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public MemberCollection(
    IEnumerable<IMember> coll
)
```

VB

```
Public Sub New (
    coll As IEnumerable(Of IMember)
)
```

Parameters

coll	Type: System.Collections.Generic.IEnumerable.IMember [▶ 1562]. The coll.

Reference

MemberCollection Class [▶ 1717]

MemberCollection Overload [▶ 1720]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.81.2 MemberCollection Properties

The MemberCollection [▶ 1717] type exposes the following members.

Properties

	Name	Description
	Count [▶ 1891]	Gets the collection count. (Inherited from InstanceCollection.T. [▶ 1887].)
	IsReadOnly [▶ 1892]	Gets a value indicating whether this instance is read only. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.Int32. [▶ 1893]	Gets or sets the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.String. [▶ 1894]	Gets the IInstance [▶ 1556] with the specified instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	Mode [▶ 1894]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

MemberCollection Class [▶ 1717]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.81.3 MemberCollection Methods

The MemberCollection [▶ 1717] type exposes the following members.

Methods

	Name	Description
	Add [▶ 1896]	Adds the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	AddRange [▶ 1897]	Adds the specified items to this collection. (Inherited from InstanceCollection.T. [▶ 1887].)
	AsReadOnly [▶ 1722]	Returns a read only copy of this collection (shallow copy)
	Clear [▶ 1898]	Clears this instance. (Inherited from InstanceCollection.T. [▶ 1887].)

	Name	Description
	Clone [▶ 1723]	Clones this MemberCollection [▶ 1717].
	Contains(String) [▶ 1899]	Determines whether this collection contains an IInstance [▶ 1556] with the specified InstanceName / InstancePath (Inherited from InstanceCollection.T. [▶ 1887].)
	Contains(T) [▶ 1899]	Determines whether this collection contains the specified IInstance [▶ 1556] (Inherited from InstanceCollection.T. [▶ 1887].)
	ContainsName [▶ 1900]	Determines whether the specified instance name contains name. (Inherited from InstanceCollection.T. [▶ 1887].)
	CopyTo [▶ 1901]	Copies this InstanceCollection.T. [▶ 1887] to the specified array. (Inherited from InstanceCollection.T. [▶ 1887].)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1902]	Gets the enumerator. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1902]	Gets the IInstance [▶ 1556]by instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetInstanceByName [▶ 1903]	Gets the name of the instance by. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1904]	Determines the index of the specified IInstance [▶ 1556]. (Inherited from InstanceCollection.T. [▶ 1887].)
	Insert [▶ 1905]	Inserts the specified IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1905]	Removes the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	RemoveAt [▶ 1906]	Removes the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1907]	Tries to get the IInstance [▶ 1556]. of the specified path. (Inherited from InstanceCollection.T. [▶ 1887].)
	TryGetInstanceByName [▶ 1908]	Tries to get Instnaces by name. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

MemberCollection Class [▶ 1717]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.81.3.1 MemberCollection.AsReadOnly Method

Returns a read only copy of this collection (shallow copy)

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyMemberCollection AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlyMemberCollection
```

Return Value

Type: ReadOnlyMemberCollection [▶ 1755]
The readonly copy.

Reference

MemberCollection Class [▶ 1717]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.81.3.2 MemberCollection.Clone Method

Clones this MemberCollection [▶ 1717].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public MemberCollection Clone()
```

VB

```
Public Function Clone As MemberCollection
```

Return Value

Type: MemberCollection [▶ 1717]
A cloned MemberCollection [▶ 1717].

Reference

MemberCollection Class [▶ 1717]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.81.4 MemberCollection Fields

The MemberCollection [▶ 1717] type exposes the following members.

Fields

	Name	Description
	_list [▶ 1909]	The _list (Inherited from InstanceCollection.T. [▶ 1887].)
	_pathDict [▶ 1909]	The _path dictionary (Inherited from InstanceCollection.T. [▶ 1887].)
	mode [▶ 1910]	The mode this InstanceCollection.T. [▶ 1887] is working in. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

MemberCollection Class [▶ 1717]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.82 MethodParamFlags Enumeration

Flag set specifying the MethodParameter context

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[FlagsAttribute]
public enum MethodParamFlags
```

VB

```
<FlagsAttribute>
Public Enumeration MethodParamFlags
```

Members

	Member name	Value	Description
	In	1	Input Parameter (ADSMETHODPARAFLAG_IN)
	Out	2	Output Parameter (ADSMETHODPARAFLAG_OUT)
	ByReference	4	By reference Parameter (ADSMETHODPARAFLAG_BYREFERENCE)
	MaskIn	5	Mask for In parameters.
	MaskOut	6	Mask for Out parameters.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.83 PrimitiveTypeFlags Enumeration

Enum PrimitiveTypeFlags

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[FlagsAttribute]
public enum PrimitiveTypeFlags
```

VB

```
<FlagsAttribute>
Public Enumeration PrimitiveTypeFlags
```

Members

	Member name	Value	Description
	None	0	None / Uninitialized
	System	1	System Type like Byte / Word / DWORD
	Unsigned	2	Primitive Type is Unsigned
	Bool	4	Boolean Value (maps to true and false)
	Float	8	Floating Point
	Date	16	Type represents a Date
	Time	32	Type represents a Time
	Numeric	64	Numeric value
	Bitset	128	Bitset
	MaskNumericUnsigned	66	Numeric / Unsigned Mask
	MaskDateTime	48	Date / Time Mask

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.84 RawValueChangedArgs Class

Event args for the RawValueChanged [▶ 1683] event.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.TypeSystem.ValueChangedBaseArgs [▶ 1850]
      TwinCAT.TypeSystem.RawValueChangedArgs

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class RawValueChangedArgs : ValueChangedBaseArgs
```

VB

```
Public Class RawValueChangedArgs
    Inherits ValueChangedBaseArgs
```

The RawValueChangedArgs type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Fields

	Name	Description
	Symbol [▶ 1852]	Symbol that caused the event. (Inherited from ValueChangedBaseArgs [▶ 1850].)
	UtcLocalSystemTime [▶ 1853]	Notification Time stamp of the local system (user/desktop time) in UTC (Inherited from ValueChangedBaseArgs [▶ 1850].)
	UtcRtime [▶ 1853]	Notification Time stamp of the Real time System in UTC (Inherited from ValueChangedBaseArgs [▶ 1850].)
	Value [▶ 1727]	New Value (byte[])

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.84.1 RawValueChangedArgs Methods

The RawValueChangedArgs [▶ 1725] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

RawValueChangedArgs Class [▶ 1725]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.84.2 RawValueChangedArgs Fields

The RawValueChangedArgs [▶ 1725] type exposes the following members.

Fields

	Name	Description
	Symbol [▶ 1852]	Symbol that caused the event. (Inherited from ValueChangedBaseArgs [▶ 1850].)
	UtcLocalSystemTime [▶ 1853]	Notification Time stamp of the local system (user/desktop time) in UTC (Inherited from ValueChangedBaseArgs [▶ 1850].)
	UtcRtime [▶ 1853]	Notification Time stamp of the Real time System in UTC (Inherited from ValueChangedBaseArgs [▶ 1850].)
	Value [▶ 1727]	New Value (byte[])

Reference

RawValueChangedArgs Class [▶ 1725]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.84.2.1 RawValueChangedArgs.Value Field

New Value (byte[])

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly byte[] Value
```

VB

```
Public ReadOnly Value As Byte()
```

Field Value

Type: .Byte.

Reference

RawValueChangedArgs Class [▶ 1725]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.85 ReadOnlyDataTypeCollection Class

ReadOnly Collection of IDataType [▶ 1517] objects.

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.IDataType [▶ 1517].
    TwinCAT.TypeSystem.Generic.ReadOnlyDataTypeCollection [▶ 1935].IDataType [▶ 1517].
      TwinCAT.TypeSystem.ReadOnlyDataTypeCollection

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyDataTypeCollection : ReadOnlyDataTypeCollection<IDataType>
```

VB

```
Public Class ReadOnlyDataTypeCollection
    Inherits ReadOnlyDataTypeCollection(Of IDataType)
```

The ReadOnlyDataTypeCollection type exposes the following members.

Constructors

	Name	Description
	ReadOnlyDataTypeCollection [▶ 1729]	Initializes a new instance of the ReadOnlyDataTypeCollection class.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)
	Item.String. [▶ 1939]	Gets the element with the specified type name. (Inherited from ReadOnlyDataTypeCollection.T. [▶ 1935].)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)

Methods

	Name	Description
	Contains	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)
	ContainsType [▶ 1940]	Determines whether the specified name contains type. (Inherited from ReadOnlyDataTypeCollection.T. [▶ 1935].)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetType [▶ 1941]	Tries to get the Type with the specified name out of the collection. (Inherited from ReadOnlyDataTypeCollection.T. [▶ 1935].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.85.1 ReadOnlyDataTypeCollection Constructor

Initializes a new instance of the ReadOnlyDataTypeCollection [▶ 1727] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyDataTypeCollection(
    DataTypeCollection<IDataType> coll
)
```

VB

```
Public Sub New (
    coll As DataTypeCollection(Of IDataType)
)
```

Parameters

coll	Type: TwinCAT.TypeSystem.Generic.DataTypeCollection [▶ 1855].IDataType [▶ 1517]. Collection of types.

Reference

ReadOnlyDataTypeCollection Class [▶ 1727]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.85.2 ReadOnlyDataTypeCollection Properties

The ReadOnlyDataTypeCollection [▶ 1727] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)
	Item.String. [▶ 1939]	Gets the element with the specified type name. (Inherited from ReadOnlyDataTypeCollection.T. [▶ 1935].)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)

Reference

ReadOnlyDataTypeCollection Class [▶ 1727]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.85.3 ReadOnlyDataTypeCollection Methods

The ReadOnlyDataTypeCollection [▶ 1727] type exposes the following members.

Methods

	Name	Description
	Contains	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)
	ContainsType [▶ 1940]	Determines whether the specified name contains type. (Inherited from ReadOnlyDataTypeCollection.T. [▶ 1935].)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IDataType [▶ 1517]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetType [▶ 1941]	Tries to get the Type with the specified name out of the collection. (Inherited from ReadOnlyDataTypeCollection.T. [▶ 1935].)

Reference

ReadOnlyDataTypeCollection Class [▶ 1727]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.86 ReadOnlyDimensionCollection Class

ReadOnly version of the DimensionCollection [▶ 1209]

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.IDimension [▶ 1525].
    TwinCAT.TypeSystem.ReadOnlyDimensionCollection

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyDimensionCollection : ReadOnlyCollection<IDimension>,
    IDimensionCollection, IList<IDimension>, ICollection<IDimension>,
    IEnumerable<IDimension>, IEnumerable
```

VB

```
Public Class ReadOnlyDimensionCollection
    Inherits ReadOnlyCollection(Of IDimension)
    Implements IDimensionCollection, IList(Of IDimension),
    ICollection(Of IDimension), IEnumerable(Of IDimension), IEnumerable
```

The ReadOnlyDimensionCollection type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)
	ElementCount [▶ 1733]	Gets the Number of elements in all Dimensions
	Item	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)
	LowerBounds [▶ 1733]	Gets the lower bounds.
	UpperBounds [▶ 1734]	Gets the upper bounds.

Methods

	Name	Description
	Contains	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)

	Name	Description
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDimensionLengths [▶ 1735]	Gets an array the specifies the Lengths of each Array Dimension
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.86.1 ReadOnlyDimensionCollection Properties

The ReadOnlyDimensionCollection [▶ 1731] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)
	ElementCount [▶ 1733]	Gets the Number of elements in all Dimensions
	Item	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)
	LowerBounds [▶ 1733]	Gets the lower bounds.
	UpperBounds [▶ 1734]	Gets the upper bounds.

Reference

ReadOnlyDimensionCollection Class [▶ 1731]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.86.1.1 ReadOnlyDimensionCollection.ElementCount Property

Gets the Number of elements in all Dimensions

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int ElementCount { get; }
```

VB

```
Public ReadOnly Property ElementCount As Integer
    Get
```

Property Value

Type: Int32

Implements

IDimensionCollection.ElementCount [▶ 1529]

Reference

ReadOnlyDimensionCollection Class [▶ 1731]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.86.1.2 ReadOnlyDimensionCollection.LowerBounds Property

Gets the lower bounds.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int[] LowerBounds { get; }
```

VB

```
Public ReadOnly Property LowerBounds As Integer()
    Get
```

Property Value

Type: .Int32.
The lower bounds.

Implements

IDimensionCollection.LowerBounds [▶ 1529]

Reference

ReadOnlyDimensionCollection Class [▶ 1731]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.86.1.3 ReadOnlyDimensionCollection.UpperBounds Property

Gets the upper bounds.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int[] UpperBounds { get; }
```

VB

```
Public ReadOnly Property UpperBounds As Integer()
    Get
```

Property Value

Type: .Int32.
The upper bounds.

Implements

IDimensionCollection.UpperBounds [▶ 1530]

Reference

ReadOnlyDimensionCollection Class [▶ 1731]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.86.2 ReadOnlyDimensionCollection Methods

The ReadOnlyDimensionCollection [▶ 1731] type exposes the following members.

Methods

	Name	Description
	Contains	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetDimensionLengths [▶ 1735]	Gets an array the specifies the Lengths of each Array Dimension
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IDimension [▶ 1525]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

ReadOnlyDimensionCollection Class [▶ 1731]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.86.2.1 ReadOnlyDimensionCollection.GetDimensionLengths Method

Gets an array the specifies the Lengths of each Array Dimension

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int[] GetDimensionLengths()
```

VB

```
Public Function GetDimensionLengths As Integer()
```

Return Value

Type: .Int32.
System.Int32[].

Implements

IDimensionCollection.GetDimensionLengths. [▶ 1531]

Reference

ReadOnlyDimensionCollection Class [▶ 1731]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.87 ReadOnlyEnumValueCollection Class

Read only version of the EnumValueCollection.T. [▶ 1467]

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.IEnumValue [▶ 1550].
    TwinCAT.TypeSystem.ReadOnlyEnumValueCollection

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyEnumValueCollection : ReadOnlyCollection<IEnumValue>
```

VB

```
Public Class ReadOnlyEnumValueCollection
    Inherits ReadOnlyCollection(Of IEnumValue)
```

The ReadOnlyEnumValueCollection type exposes the following members.

Constructors

	Name	Description
	ReadOnlyEnumValueCollection [▶ 1737]	Initializes a new instance of the ReadOnlyEnumValueCollection.T. [▶ 1743] class.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)
	Item.String. [▶ 1738]	Gets or sets the element at the specified index.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)

Methods

	Name	Description
	Contains(String) [▶ 1740]	Determines whether [contains] [the specified name].
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetNames [▶ 1741]	Gets the Value Names.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	GetValues [▶ 1741]	Gets the values.
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Parse [▶ 1742]	Parses the specified name.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryParse [▶ 1742]	Tries to pars the string value of the Enum.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.87.1 ReadOnlyEnumValueCollection Constructor

Initializes a new instance of the ReadOnlyEnumValueCollection.T. [▶ 1743] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyEnumValueCollection(
    EnumValueCollection coll
)
```

VB

```
Public Sub New (
    coll As EnumValueCollection
)
```

Parameters

coll	Type: TwinCAT.TypeSystem.EnumValueCollection [▶ 1450] The coll.

Reference

ReadOnlyEnumValueCollection Class [▶ 1736]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.87.2 ReadOnlyEnumValueCollection Properties

The ReadOnlyEnumValueCollection [▶ 1736] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)
	Item.String. [▶ 1738]	Gets or sets the element at the specified index.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)

Reference

ReadOnlyEnumValueCollection Class [▶ 1736]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.87.2.1 ReadOnlyEnumValueCollection.Item Property

Overload List

	Name	Description
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)
	Item.String. [▶ 1738]	Gets or sets the element at the specified index.

Reference

ReadOnlyEnumValueCollection Class [▶ 1736]

TwinCAT.TypeSystem Namespace [▶ 1184]

ReadOnlyEnumValueCollection.Item Property (String)

Gets or sets the element at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumValue this[
    string name
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    name As String
) As IEnumValue
    Get
```

Parameters

name	Type: System.String The name of the value

Return Value

Type: IEnumValue [▶ 1550]
EnumValue<T>.

Exceptions

Exception	Condition
NotImplementedException	
NotImplementedException	

Reference

ReadOnlyEnumValueCollection Class [▶ 1736]

Item Overload [▶ 1738]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.87.3 ReadOnlyEnumValueCollection Methods

The ReadOnlyEnumValueCollection [▶ 1736] type exposes the following members.

Methods

	Name	Description
	Contains(String) [▶ 1740]	Determines whether [contains] [the specified name].
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)

	Name	Description
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetNames [▶ 1741]	Gets the Value Names.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	GetValues [▶ 1741]	Gets the values.
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Parse [▶ 1742]	Parses the specified name.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryParse [▶ 1742]	Tries to pars the string value of the Enum.

Reference

ReadOnlyEnumValueCollection Class [▶ 1736]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.87.3.1 ReadOnlyEnumValueCollection.Contains Method

Overload List

	Name	Description
	Contains(String) [▶ 1740]	Determines whether [contains] [the specified name].
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IEnumValue [▶ 1550]..)

Reference

ReadOnlyEnumValueCollection Class [▶ 1736]

TwinCAT.TypeSystem Namespace [▶ 1184]

ReadOnlyEnumValueCollection.Contains Method (String)

Determines whether [contains] [the specified name].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    string value
)
```

VB

```
Public Function Contains (
    value As String
) As Boolean
```

Parameters

value	Type: System.String Value

Return Value

Type: Boolean
true if [contains] [the specified name]; otherwise, false.

Reference

ReadOnlyEnumValueCollection Class [▶ 1736]

Contains Overload [▶ 1740]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.87.3.2 ReadOnlyEnumValueCollection.GetNames Method

Gets the Value Names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string[] GetNames()
```

VB

```
Public Function GetNames As String()
```

Return Value

Type: .String.
System.String[].

Reference

ReadOnlyEnumValueCollection Class [▶ 1736]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.87.3.3 ReadOnlyEnumValueCollection.GetValues Method

Gets the values.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object[] GetValues()
```

VB

```
Public Function GetValues As Object()
```

Return Value

Type: .Object.
T[].

Reference

ReadOnlyEnumValueCollection Class [▶ 1736]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.87.3.4 ReadOnlyEnumValueCollection.Parse Method

Parses the specified name.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public Object Parse(
    string name
)
```

VB

```
Public Function Parse (
    name As String
) As Object
```

Parameters

name	Type: System.String The name.

Return Value

Type: Object
T.

Reference

ReadOnlyEnumValueCollection Class [▶ 1736]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.87.3.5 ReadOnlyEnumValueCollection.TryParse Method

Tries to pars the string value of the Enum.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryParse(
    string strValue,
    Object value
)
```

VB

```
Public Function TryParse (
    strValue As String,
    value As Object
) As Boolean
```

Parameters

strValue	Type: System.String The Value in string represention.
value	Type: System.Object The value.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

ReadOnlyEnumValueCollection Class [▶ 1736]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.88 ReadOnlyEnumValueCollection.T. Class

Read only version of the EnumValueCollection.T. [▶ 1467]

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.EnumValue [▶ 1443].T..
    TwinCAT.TypeSystem.ReadOnlyEnumValueCollection.T.
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyEnumValueCollection<T> : ReadOnlyCollection<EnumValue<T>>
where T : IConvertible
```

VB

```
Public Class ReadOnlyEnumValueCollection(Of T As IConvertible)
    Inherits ReadOnlyCollection(Of EnumValue(Of T))
```

Type Parameters

T

The ReadOnlyEnumValueCollection.T. type exposes the following members.

Constructors

	Name	Description
	ReadOnlyEnumValueCollection.T. [▶ 1745]	Initializes a new instance of the ReadOnlyEnumValueCollection.T. class.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T...)
	Item	Gets the element at the specified index. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T...)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T...)

Methods

	Name	Description
	Contains(String) [▶ 1747]	Determines whether [contains] [the specified name].
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T...)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T...)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T...)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetNames [▶ 1747]	Gets the Value Names.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	GetValues [▶ 1748]	Gets the values.
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T...)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Parse [▶ 1748]	Parses the specified name.

	Name	Description
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryParse [▶ 1749]	Tries to parse the string value of the Enum.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.88.1 ReadOnlyEnumValueCollection.T. Constructor

Initializes a new instance of the ReadOnlyEnumValueCollection.T. [▶ 1743] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyEnumValueCollection(
    EnumValueCollection<T> coll
)
```

VB

```
Public Sub New (
    coll As EnumValueCollection(Of T)
)
```

Parameters

coll	Type: TwinCAT.TypeSystem.EnumValueCollection [▶ 1467].T [▶ 1743]. The coll.

Reference

ReadOnlyEnumValueCollection.T. Class [▶ 1743]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.88.2 ReadOnlyEnumValueCollection.T. Properties

The ReadOnlyEnumValueCollection.T. [▶ 1743] generic type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T [▶ 1743]...)
	Item	Gets the element at the specified index. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T [▶ 1743]...)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T [▶ 1743]...)

Reference

ReadOnlyEnumValueCollection.T. Class [▶ 1743]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.88.3 ReadOnlyEnumValueCollection.T. Methods

The ReadOnlyEnumValueCollection.T. [▶ 1743] generic type exposes the following members.

Methods

	Name	Description
	Contains(String) [▶ 1747]	Determines whether [contains] [the specified name].
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T [▶ 1743]...)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T [▶ 1743]...)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T [▶ 1743]...)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetNames [▶ 1747]	Gets the Value Names.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	GetValues [▶ 1748]	Gets the values.
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T [▶ 1743]...)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Parse [▶ 1748]	Parses the specified name.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryParse [▶ 1749]	Tries to parse the string value of the Enum.

Reference

ReadOnlyEnumValueCollection.T. Class [▶ 1743]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.88.3.1 ReadOnlyEnumValueCollection.T..Contains Method

Overload List

	Name	Description
	Contains(String) [▶ 1747]	Determines whether [contains] [the specified name].
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.EnumValue [▶ 1443].T [▶ 1743]...)

Reference

ReadOnlyEnumValueCollection.T. Class [▶ 1743]

TwinCAT.TypeSystem Namespace [▶ 1184]

ReadOnlyEnumValueCollection.T..Contains Method (String)

Determines whether [contains] [the specified name].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    string value
)
```

VB

```
Public Function Contains (
    value As String
) As Boolean
```

Parameters

value	Type: System.String Value	

Return Value

Type: Boolean
true if [contains] [the specified name]; otherwise, false.

Reference

ReadOnlyEnumValueCollection.T. Class [▶ 1743]

Contains Overload [▶ 1747]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.88.3.2 ReadOnlyEnumValueCollection.T..GetNames Method

Gets the Value Names.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string[] GetNames()
```

VB

```
Public Function GetNames As String()
```

Return Value

Type: .String.
System.String[].

Reference

ReadOnlyEnumValueCollection.T. Class [▶ 1743]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.88.3.3 ReadOnlyEnumValueCollection.T..GetValues Method

Gets the values.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T[] GetValues()
```

VB

```
Public Function GetValues As T()
```

Return Value

Type: .T [▶ 1743].
T[].

Reference

ReadOnlyEnumValueCollection.T. Class [▶ 1743]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.88.3.4 ReadOnlyEnumValueCollection.T..Parse Method

Parses the specified name.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T Parse(
    string name
)
```

VB

```
Public Function Parse (
    name As String
) As T
```

Parameters

name	Type: System.String The name.

Return Value

Type: T [▶ 1743]
T.

Reference

ReadOnlyEnumValueCollection.T. Class [▶ 1743]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.88.3.5 ReadOnlyEnumValueCollection.T..TryParse Method

Tries to parse the string value of the Enum.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryParse(
    string strValue,
    out T value
)
```

VB

```
Public Function TryParse (
    strValue As String,
    <OutAttribute> ByRef value As T
) As Boolean
```

Parameters

strValue	Type: System.String The Value in string represention.
value	Type: T [▶ 1743]. The value.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

ReadOnlyEnumValueCollection.T. Class [▶ 1743]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.89 ReadOnlyFieldCollection Class

Read only collection of IField [▶ 1554] objects

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.IField [▶ 1554].
    TwinCAT.TypeSystem.Generic.ReadOnlyInstanceCollection [▶ 1942].IField [▶ 1554].
      TwinCAT.TypeSystem.ReadOnlyFieldCollection
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyFieldCollection : ReadOnlyInstanceCollection<IField>
```

VB

```
Public Class ReadOnlyFieldCollection
    Inherits ReadOnlyInstanceCollection(Of IField)
```

The ReadOnlyFieldCollection type exposes the following members.

Constructors

	Name	Description
	ReadOnlyFieldCollection [▶ 1752]	Initializes a new instance of the ReadOnlyMemberCollection [▶ 1755] class.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	Item.String. [▶ 1945]	Gets the element with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	Mode [▶ 1946]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Methods

	Name	Description
	Contains(String) [▶ 1947]	Determines whether the ReadOnlyInstanceCollection.T. [▶ 1942] contains an instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	ContainsName [▶ 1948]	Determines whether the specified instance is contained. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1949]	Gets the IInstance [▶ 1556]by instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetInstanceByName [▶ 1950]	Gets the IInstance [▶ 1556] by instance name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1950]	Tries to get the instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	TryGetInstanceByName [▶ 1951]	Tries to get the instance by name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	TryGetMember [▶ 1754]	Tries to get the specified member

Fields

	Name	Description
	mode [▶ 1952]	Mode of the IInstanceCollection.T. [▶ 1877] (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.89.1 ReadOnlyFieldCollection Constructor

Initializes a new instance of the ReadOnlyMemberCollection [▶ 1755] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyFieldCollection(
    FieldCollection members
)
```

VB

```
Public Sub New (
    members As FieldCollection
)
```

Parameters

members	Type: TwinCAT.TypeSystem.FieldCollection [▶ 1484] The members.

Reference

ReadOnlyFieldCollection Class [▶ 1750]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.89.2 ReadOnlyFieldCollection Properties

The ReadOnlyFieldCollection [▶ 1750] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	Item.String. [▶ 1945]	Gets the element with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	Mode [▶ 1946]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

ReadOnlyFieldCollection Class [▶ 1750]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.89.3 ReadOnlyFieldCollection Methods

The ReadOnlyFieldCollection [▶ 1750] type exposes the following members.

Methods

	Name	Description
	Contains(String) [▶ 1947]	Determines whether the ReadOnlyInstanceCollection.T. [▶ 1942] contains an instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	ContainsName [▶ 1948]	Determines whether the specified instance is contained. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1949]	Gets the IInstance [▶ 1556]by instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetInstanceByName [▶ 1950]	Gets the IInstance [▶ 1556] by instance name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IField [▶ 1554]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1950]	Tries to get the instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	TryGetInstanceByName [▶ 1951]	Tries to get the instance by name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	TryGetMember [▶ 1754]	Tries to get the specified member

Reference

ReadOnlyFieldCollection Class [▶ 1750]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.89.3.1 ReadOnlyFieldCollection.TryGetMember Method

Tries to get the specified member

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetMember(
    string fieldName,
    out IField symbol
)
```

VB

```
Public Function TryGetMember (
    fieldName As String,
    <OutAttribute> ByRef symbol As IField
) As Boolean
```

Parameters

fieldName	Type: System.String Name of the member.
symbol	Type: TwinCAT.TypeSystem.IField [▶ 1554]. The symbol.

Return Value

Type: Boolean
true if found, false otherwise.

Reference

ReadOnlyFieldCollection Class [▶ 1750]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.89.4 ReadOnlyFieldCollection Fields

The ReadOnlyFieldCollection [▶ 1750] type exposes the following members.

Fields

	Name	Description
	mode [▶ 1952]	Mode of the IInstanceCollection.T. [▶ 1877] (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

ReadOnlyFieldCollection Class [▶ 1750]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.90 ReadOnlyMemberCollection Class

Read only collection of IMember [▶ 1562] objects

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.IMember [▶ 1562].
    TwinCAT.TypeSystem.Generic.ReadOnlyInstanceCollection [▶ 1942].IMember [▶ 1562].
      TwinCAT.TypeSystem.ReadOnlyMemberCollection

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyMemberCollection : ReadOnlyInstanceCollection<IMember>
```

VB

```
Public Class ReadOnlyMemberCollection
    Inherits ReadOnlyInstanceCollection(Of IMember)
```

The ReadOnlyMemberCollection type exposes the following members.

Constructors

	Name	Description
	ReadOnlyMemberCollection [▶ 1756]	Initializes a new instance of the ReadOnlyMemberCollection class.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	Item.String. [▶ 1945]	Gets the element with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	Mode [▶ 1946]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Methods

	Name	Description
	Contains(String) [▶ 1947]	Determines whether the ReadOnlyInstanceCollection.T. [▶ 1942] contains an instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)

	Name	Description
	ContainsName [▶ 1948]	Determines whether the specified instance is contained. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1949]	Gets the IInstance [▶ 1556]by instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetInstanceByName [▶ 1950]	Gets the IInstance [▶ 1556] by instance name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1950]	Tries to get the instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	TryGetInstanceByName [▶ 1951]	Tries to get the instance by name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	TryGetMember [▶ 1758]	Tries to get the specified member

Fields

	Name	Description
	mode [▶ 1952]	Mode of the IInstanceCollection.T. [▶ 1877] (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.90.1 ReadOnlyMemberCollection Constructor

Initializes a new instance of the ReadOnlyMemberCollection [▶ 1755] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyMemberCollection(
    MemberCollection members
)
```

VB

```
Public Sub New (
    members As MemberCollection
)
```

Parameters

members	Type: TwinCAT.TypeSystem.MemberCollection [▶ 1717] The members.

Reference

ReadOnlyMemberCollection Class [▶ 1755]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.90.2 ReadOnlyMemberCollection Properties

The ReadOnlyMemberCollection [▶ 1755] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	Item.String. [▶ 1945]	Gets the element with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	Mode [▶ 1946]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

ReadOnlyMemberCollection Class [▶ 1755]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.90.3 ReadOnlyMemberCollection Methods

The ReadOnlyMemberCollection [▶ 1755] type exposes the following members.

Methods

	Name	Description
	Contains(String) [▶ 1947]	Determines whether the ReadOnlyInstanceCollection.T. [▶ 1942] contains an instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	ContainsName [▶ 1948]	Determines whether the specified instance is contained. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1949]	Gets the IInstance [▶ 1556]by instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetInstanceByName [▶ 1950]	Gets the IInstance [▶ 1556] by instance name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IMember [▶ 1562]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1950]	Tries to get the instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	TryGetInstanceByName [▶ 1951]	Tries to get the instance by name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	TryGetMember [▶ 1758]	Tries to get the specified member

Reference

ReadOnlyMemberCollection Class [▶ 1755]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.90.3.1 ReadOnlyMemberCollection.TryGetMember Method

Tries to get the specified member

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetMember(
    string memberName,
    out IMember symbol
)
```

VB

```
Public Function TryGetMember (
    memberName As String,
    <OutAttribute> ByRef symbol As IMember
) As Boolean
```

Parameters

memberName	Type: System.String Name of the member.
symbol	Type: TwinCAT.TypeSystem.IMember [▶ 1562]. The symbol.

Return Value

Type: Boolean
true if found, false otherwise.

Reference

ReadOnlyMemberCollection Class [▶ 1755]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.90.4 ReadOnlyMemberCollection Fields

The ReadOnlyMemberCollection [▶ 1755] type exposes the following members.

Fields

	Name	Description
	mode [▶ 1952]	Mode of the IInstanceCollection.T. [▶ 1877] (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

ReadOnlyMemberCollection Class [▶ 1755]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.91 ReadOnlyMethodParameterCollection Class

Read only RpcMethodParameterCollection [▶ 1797].

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.IRpcMethodParameter [▶ 1601].
    TwinCAT.TypeSystem.ReadOnlyMethodParameterCollection

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyMethodParameterCollection : ReadOnlyCollection<IRpcMethodParameter>
```

VB

```
Public Class ReadOnlyMethodParameterCollection
    Inherits ReadOnlyCollection(Of IRpcMethodParameter)
```

The ReadOnlyMethodParameterCollection type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)
	Item	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)

Methods

	Name	Description
	Contains	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.91.1 ReadOnlyMethodParameterCollection Properties

The ReadOnlyMethodParameterCollection [▶ 1759] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)
	Item	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)

Reference

ReadOnlyMethodParameterCollection Class [▶ 1759]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.91.2 ReadOnlyMethodParameterCollection Methods

The ReadOnlyMethodParameterCollection [▶ 1759] type exposes the following members.

Methods

	Name	Description
	Contains	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethodParameter [▶ 1601]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

ReadOnlyMethodParameterCollection Class [▶ 1759]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.92 ReadOnlyRpcMethodCollection Class

Read only RpcMethodCollection [▶ 1782]

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.IRpcMethod [▶ 1597].
    TwinCAT.TypeSystem.ReadOnlyRpcMethodCollection

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyRpcMethodCollection : ReadOnlyCollection<IRpcMethod>
```

VB

```
Public Class ReadOnlyRpcMethodCollection
    Inherits ReadOnlyCollection(Of IRpcMethod)
```

The ReadOnlyRpcMethodCollection type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	Item.String. [▶ 1764]	Gets the IRpcMethod [▶ 1597] with the specified method name.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)

Methods

	Name	Description
	Contains(String) [▶ 1766]	Determines whether this collection contains the specified method name.
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)

	Name	Description
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetMethod(Int32, IRpcMethod.) [▶ 1767]	Tries to get the specified method.
	TryGetMethod(String, IRpcMethod.) [▶ 1767]	Tries to get the specified method.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.92.1 ReadOnlyRpcMethodCollection Properties

The ReadOnlyRpcMethodCollection [▶ 1762] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	Item.String. [▶ 1764]	Gets the IRpcMethod [▶ 1597] with the specified method name.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)

Reference

ReadOnlyRpcMethodCollection Class [▶ 1762]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.92.1.1 ReadOnlyRpcMethodCollection.Item Property

Overload List

	Name	Description
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	Item.String. [▶ 1764]	Gets the IRpcMethod [▶ 1597] with the specified method name.

Reference

ReadOnlyRpcMethodCollection Class [▶ 1762]

TwinCAT.TypeSystem Namespace [▶ 1184]

ReadOnlyRpcMethodCollection.Item Property (String)

Gets the IRpcMethod [▶ 1597] with the specified method name.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IRpcMethod this[
    string methodName
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    methodName As String
) As IRpcMethod
    Get
```

Parameters

methodName	Type: System.String Name of the method.

Return Value

Type: IRpcMethod [▶ 1597]
RpcMethod.

Exceptions

Exception	Condition
KeyNotFoundException	

Reference

ReadOnlyRpcMethodCollection Class [▶ 1762]

Item Overload [▶ 1764]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.92.2 ReadOnlyRpcMethodCollection Methods

The ReadOnlyRpcMethodCollection [▶ 1762] type exposes the following members.

Methods

	Name	Description
	Contains(String) [▶ 1766]	Determines whether this collection contains the specified method name.
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetMethod(Int32, IRpcMethod.) [▶ 1767]	Tries to get the specified method.
	TryGetMethod(String, IRpcMethod.) [▶ 1767]	Tries to get the specified method.

Reference

ReadOnlyRpcMethodCollection Class [▶ 1762]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.92.2.1 ReadOnlyRpcMethodCollection.Contains Method

Overload List

	Name	Description
	Contains(String) [▶ 1766]	Determines whether this collection contains the specified method name.

	Name	Description
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.IRpcMethod [▶ 1597]..)

Reference

ReadOnlyRpcMethodCollection Class [▶ 1762]

TwinCAT.TypeSystem Namespace [▶ 1184]

ReadOnlyRpcMethodCollection.Contains Method (String)

Determines whether this collection contains the specified method name.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    string methodName
)
```

VB

```
Public Function Contains (
    methodName As String
) As Boolean
```

Parameters

methodName	Type: System.String Name of the method.

Return Value

Type: Boolean
true if contained.; otherwise, false.

Reference

ReadOnlyRpcMethodCollection Class [▶ 1762]

Contains Overload [▶ 1765]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.92.2.2 ReadOnlyRpcMethodCollection.TryGetMethod Method

Overload List

	Name	Description
	TryGetMethod(Int32, IRpcMethod.) [▶ 1767]	Tries to get the specified method.

	Name	Description
	TryGetMethod(String, IRpcMethod.) [▶ 1767]	Tries to get the specified method.

Reference

ReadOnlyRpcMethodCollection Class [▶ 1762]

TwinCAT.TypeSystem Namespace [▶ 1184]

ReadOnlyRpcMethodCollection.TryGetMethod Method (Int32, IRpcMethod.)

Tries to get the specified method.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetMethod(
    int vTableIndex,
    out IRpcMethod method
)
```

VB

```
Public Function TryGetMethod (
    vTableIndex As Integer,
    <OutAttribute> ByRef method As IRpcMethod
) As Boolean
```

Parameters

vTableIndex	Type: System.Int32 vTableIndex.
method	Type: TwinCAT.TypeSystem.IRpcMethod [▶ 1597]. The method if fund, NULL otherwise.

Return Value

Type: Boolean
true if found, false otherwise.

Reference

ReadOnlyRpcMethodCollection Class [▶ 1762]

TryGetMethod Overload [▶ 1766]

TwinCAT.TypeSystem Namespace [▶ 1184]

ReadOnlyRpcMethodCollection.TryGetMethod Method (String, IRpcMethod.)

Tries to get the specified method.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetMethod(
    string methodName,
    out IRpcMethod method
)
```

VB

```
Public Function TryGetMethod (
    methodName As String,
    <OutAttribute> ByRef method As IRpcMethod
) As Boolean
```

Parameters

methodName	Type: System.String Name of the method.
method	Type: TwinCAT.TypeSystem.IRpcMethod [▶ 1597]. The method if fund, NULL otherwise.

Return Value

Type: Boolean
true if found, false otherwise.

Reference

ReadOnlyRpcMethodCollection Class [▶ 1762]

TryGetMethod Overload [▶ 1766]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.93 ReadOnlySubItemCollection Class

Class ReadOnlySubItemCollection.

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.ITcAdsSubItem [▶ 544].
    TwinCAT.TypeSystem.ReadOnlySubItemCollection
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlySubItemCollection : ReadOnlyCollection<ITcAdsSubItem>
```

VB

```
Public Class ReadOnlySubItemCollection
    Inherits ReadOnlyCollection(Of ITcAdsSubItem)
```

The ReadOnlySubItemCollection type exposes the following members.

Constructors

	Name	Description
	ReadOnlySubItemCollection. [▶ 1770]	Initializes a new instance of the ReadOnlySubItemCollection class.
	ReadOnlySubItemCollection(IList.ITcAdsSubItem.) [▶ 1770]	Initializes a new instance of the ReadOnlySubItemCollection class.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)
	Item	Gets the element at the specified index. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)

Methods

	Name	Description
	Contains	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.93.1 ReadOnlySubItemCollection Constructor

Overload List

	Name	Description
	ReadOnlySubItemCollection. [▶ 1770]	Initializes a new instance of the ReadOnlySubItemCollection [▶ 1768] class.
	ReadOnlySubItemCollection(IList.ITcAdsSubItem.) [▶ 1770]	Initializes a new instance of the ReadOnlySubItemCollection [▶ 1768] class.

Reference

ReadOnlySubItemCollection Class [▶ 1768]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.93.1.1 ReadOnlySubItemCollection Constructor

Initializes a new instance of the ReadOnlySubItemCollection [▶ 1768] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlySubItemCollection()
```

VB

```
Public Sub New
```

Reference

ReadOnlySubItemCollection Class [▶ 1768]

ReadOnlySubItemCollection Overload [▶ 1770]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.93.1.2 ReadOnlySubItemCollection Constructor (IList.ITcAdsSubItem.)

Initializes a new instance of the ReadOnlySubItemCollection [▶ 1768] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlySubItemCollection(
    IList<ITcAdsSubItem> coll
)
```

VB

```
Public Sub New (
    coll As IList(Of ITcAdsSubItem)
)
```

Parameters

coll	Type: System.Collections.Generic.IList.ITcAdsSubItem [▶ 544]. The coll.

Reference

ReadOnlySubItemCollection Class [▶ 1768]

ReadOnlySubItemCollection Overload [▶ 1770]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.93.2 ReadOnlySubItemCollection Properties

The ReadOnlySubItemCollection [▶ 1768] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)
	Item	Gets the element at the specified index. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)

Reference

ReadOnlySubItemCollection Class [▶ 1768]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.93.3 ReadOnlySubItemCollection Methods

The ReadOnlySubItemCollection [▶ 1768] type exposes the following members.

Methods

	Name	Description
	Contains	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITcAdsSubItem [▶ 544]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

ReadOnlySubItemCollection Class [▶ 1768]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.94 ReadOnlySymbolCollection Class

ReadOnly collection containing ISymbol [▶ 1634] objects.

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.ISymbol [▶ 1634].
    TwinCAT.TypeSystem.Generic.ReadOnlyInstanceCollection [▶ 1942].ISymbol [▶ 1634].
      TwinCAT.TypeSystem.Generic.ReadOnlySymbolCollection [▶ 1960].ISymbol [▶ 1634].
        TwinCAT.TypeSystem.ReadOnlySymbolCollection

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlySymbolCollection : ReadOnlySymbolCollection<ISymbol>,
    ISymbolCollection, ISymbolCollection<ISymbol>, IInstanceCollection<ISymbol>,
    IList<ISymbol>, ICollection<ISymbol>, IEnumerable<ISymbol>,
    IEnumerable
```

VB

```
Public Class ReadOnlySymbolCollection
    Inherits ReadOnlySymbolCollection(Of ISymbol)
    Implements ISymbolCollection, ISymbolCollection(Of ISymbol),
    IInstanceCollection(Of ISymbol), IList(Of ISymbol), ICollection(Of ISymbol),
    IEnumerable(Of ISymbol), IEnumerable
```

The ReadOnlySymbolCollection type exposes the following members.

Constructors

	Name	Description
	ReadOnlySymbolCollection [▶ 1774]	Initializes a new instance of the ReadOnlySymbolCollection class.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	Item.String. [▶ 1945]	Gets the element with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	Mode [▶ 1946]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Methods

	Name	Description
	Contains(String) [▶ 1947]	Determines whether the ReadOnlyInstanceCollection.T. [▶ 1942] contains an instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	ContainsName [▶ 1948]	Determines whether the specified instance is contained. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1949]	Gets the IInstance [▶ 1556]by instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetInstanceByName [▶ 1950]	Gets the IInstance [▶ 1556] by instance name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1950]	Tries to get the instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

	Name	Description
	TryGetInstanceByName [▶ 1951]	Tries to get the instance by name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Fields

	Name	Description
	mode [▶ 1952]	Mode of the IInstanceCollection.T. [▶ 1877] (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.94.1 ReadOnlySymbolCollection Constructor

Initializes a new instance of the ReadOnlySymbolCollection [▶ 1772] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlySymbolCollection(
    IInstanceCollection<ISymbol> symbols
)
```

VB

```
Public Sub New (
    symbols As IInstanceCollection(Of ISymbol)
)
```

Parameters

symbols	Type: TwinCAT.TypeSystem.Generic.IInstanceCollection [▶ 1877].ISymbol [▶ 1634]. The symbols.

Reference

ReadOnlySymbolCollection Class [▶ 1772]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.94.2 ReadOnlySymbolCollection Properties

The ReadOnlySymbolCollection [▶ 1772] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)

	Name	Description
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	Item.String. [▶ 1945]	Gets the element with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	Mode [▶ 1946]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

ReadOnlySymbolCollection Class [▶ 1772]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.94.3 ReadOnlySymbolCollection Methods

The ReadOnlySymbolCollection [▶ 1772] type exposes the following members.

Methods

	Name	Description
	Contains(String) [▶ 1947]	Determines whether the ReadOnlyInstanceCollection.T. [▶ 1942] contains an instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	ContainsName [▶ 1948]	Determines whether the specified instance is contained. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1949]	Gets the IInstance [▶ 1556]by instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetInstanceByName [▶ 1950]	Gets the IInstance [▶ 1556] by instance name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ISymbol [▶ 1634]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)

	Name	Description
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1950]	Tries to get the instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	TryGetInstanceByName [▶ 1951]	Tries to get the instance by name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

ReadOnlySymbolCollection Class [▶ 1772]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.94.4 ReadOnlySymbolCollection Fields

The ReadOnlySymbolCollection [▶ 1772] type exposes the following members.

Fields

	Name	Description
	mode [▶ 1952]	Mode of the IInstanceCollection.T. [▶ 1877] (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

ReadOnlySymbolCollection Class [▶ 1772]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.95 ReadOnlyTypeAttributeCollection Class

Read only version of the TypeAttributeCollection [▶ 1829]

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.ITypeAttribute [▶ 1651].
    TwinCAT.TypeSystem.ReadOnlyTypeAttributeCollection
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyTypeAttributeCollection : ReadOnlyCollection<ITypeAttribute>
```

VB

```
Public Class ReadOnlyTypeAttributeCollection
    Inherits ReadOnlyCollection(Of ITypeAttribute)
```

The ReadOnlyTypeAttributeCollection type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	Item.String. [▶ 1778]	Gets the String with the specified name.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)

Methods

	Name	Description
	Contains(String) [▶ 1780]	Determines whether this ReadOnlyTypeAttributeCollection contains the specified attribute.
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetAttribute [▶ 1781]	Tries to get the specified ITypeAttribute [▶ 1651]
	TryGetValue [▶ 1781]	Tries to get the specified Attribute value.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.95.1 ReadOnlyTypeAttributeCollection Properties

The ReadOnlyTypeAttributeCollection [▶ 1776] type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	Item.String. [▶ 1778]	Gets the String with the specified name.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)

Reference

ReadOnlyTypeAttributeCollection Class [▶ 1776]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.95.1.1 ReadOnlyTypeAttributeCollection.Item Property

Overload List

	Name	Description
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	Item.String. [▶ 1778]	Gets the String with the specified name.

Reference

ReadOnlyTypeAttributeCollection Class [▶ 1776]

TwinCAT.TypeSystem Namespace [▶ 1184]

ReadOnlyTypeAttributeCollection.Item Property (String)

Gets the String with the specified name.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string this[
    string name
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    name As String
) As String
    Get
```

Parameters

name	Type: System.String The name.

Return Value

Type: String
System.String.

Reference

ReadOnlyTypeAttributeCollection Class [▶ 1776]

Item Overload [▶ 1778]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.95.2 ReadOnlyTypeAttributeCollection Methods

The ReadOnlyTypeAttributeCollection [▶ 1776] type exposes the following members.

Methods

	Name	Description
	Contains(String) [▶ 1780]	Determines whether this ReadOnlyTypeAttributeCollection [▶ 1776] contains the specified attribute.
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetAttribute [▶ 1781]	Tries to get the specified ITypeAttribute [▶ 1651]
	TryGetValue [▶ 1781]	Tries to get the specified Attribute value.

Reference

ReadOnlyTypeAttributeCollection Class [▶ 1776]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.95.2.1 ReadOnlyTypeAttributeCollection.Contains Method

Overload List

	Name	Description
	Contains(String) [▶ 1780]	Determines whether this ReadOnlyTypeAttributeCollection [▶ 1776] contains the specified attribute.
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.ITypeAttribute [▶ 1651]..)

Reference

ReadOnlyTypeAttributeCollection Class [▶ 1776]

TwinCAT.TypeSystem Namespace [▶ 1184]

ReadOnlyTypeAttributeCollection.Contains Method (String)

Determines whether this ReadOnlyTypeAttributeCollection [▶ 1776] contains the specified attribute.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    string name
)
```

VB

```
Public Function Contains (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name.

Return Value

Type: Boolean
true if [contains] [the specified name]; otherwise, false.

Reference

ReadOnlyTypeAttributeCollection Class [▶ 1776]

Contains Overload [▶ 1780]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.95.2.2 ReadOnlyTypeAttributeCollection.TryGetAttribute Method

Tries to get the specified ITypeAttribute [▶ 1651]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetAttribute(
    string name,
    out ITypeAttribute attribute
)
```

VB

```
Public Function TryGetAttribute (
    name As String,
    <OutAttribute> ByRef attribute As ITypeAttribute
) As Boolean
```

Parameters

name	Type: System.String The name of the ITypeAttribute [▶ 1651].
attribute	Type: TwinCAT.TypeSystem.ITypeAttribute [▶ 1651]. The attribute.

Return Value

Type: Boolean
true if found, false otherwise.

Reference

ReadOnlyTypeAttributeCollection Class [▶ 1776]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.95.2.3 ReadOnlyTypeAttributeCollection.TryGetValue Method

Tries to get the specified Attribute value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetValue(
    string name,
    out string value
)
```

VB

```
Public Function TryGetValue (
    name As String,
    <OutAttribute> ByRef value As String
) As Boolean
```

Parameters

name	Type: System.String The name.
value	Type: System.String. The value.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

ReadOnlyTypeAttributeCollection Class [▶ 1776]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96 RpcMethodCollection Class

Collection of RpcMethods. [▶ 1597]

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.RpcMethodCollection
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class RpcMethodCollection : IList<IRpcMethod>,
    ICollection<IRpcMethod>, IEnumerable<IRpcMethod>, IEnumerable
```

VB

```
Public Class RpcMethodCollection
    Implements IList(Of IRpcMethod), ICollection(Of IRpcMethod),
    IEnumerable(Of IRpcMethod), IEnumerable
```

The RpcMethodCollection type exposes the following members.

Properties

	Name	Description
	Count [▶ 1784]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1784]	Gets a value indicating whether the ICollection.T. is read-only.
	Item.Int32. [▶ 1785]	Gets or sets the element at the specified index.

	Name	Description
	Item.String. [▶ 1786]	Gets the IRpcMethod [▶ 1597] with the specified method name.

Methods

	Name	Description
	Add [▶ 1788]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1789]	Gets a read only collection of this RpcMethodCollection
	Clear [▶ 1789]	Removes all items from the ICollection.T..
	Contains(String) [▶ 1790]	Determines whether this collection contains the specified method name.
	Contains(IRpcMethod) [▶ 1790]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1791]	Copies to.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1792]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1792]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1793]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1794]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1795]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetMethod(Int32, IRpcMethod.) [▶ 1796]	Tries to get the specified method.
	TryGetMethod(String, IRpcMethod.) [▶ 1796]	Tries to get the specified method.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.1 RpcMethodCollection Properties

The RpcMethodCollection [▶ 1782] type exposes the following members.

Properties

	Name	Description
	Count [▶ 1784]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1784]	Gets a value indicating whether the ICollection.T. is read-only.
	Item.Int32. [▶ 1785]	Gets or sets the element at the specified index.
	Item.String. [▶ 1786]	Gets the IRpcMethod [▶ 1597] with the specified method name.

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.1.1 RpcMethodCollection.Count Property

Gets the number of elements contained in the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Count { get; }
```

VB

```
Public ReadOnly Property Count As Integer
    Get
```

Property Value

Type: Int32
The count.

Implements

ICollection.T..Count

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.1.2 RpcMethodCollection.IsReadOnly Property

Gets a value indicating whether the ICollection.T. is read-only.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is read only; otherwise, false.

Implements

ICollection.T..IsReadOnly

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.1.3 RpcMethodCollection.Item Property

Overload List

	Name	Description
	Item.Int32. [▶ 1785]	Gets or sets the element at the specified index.
	Item.String. [▶ 1786]	Gets the IRpcMethod [▶ 1597] with the specified method name.

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

RpcMethodCollection.Item Property (Int32)

Gets or sets the element at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IRpcMethod this[
    int index
] { get; set; }
```

VB

```
Public Default Property Item (
    index As Integer
) As IRpcMethod
    Get
    Set
```

Parameters

index	Type: System.Int32 The index.

Return Value

Type: IRpcMethod [▶ 1597]
RpcMethod.

Implements

IList.T..Item.Int32.

Exceptions

Exception	Condition
NotImplementedException	

Reference

RpcMethodCollection Class [▶ 1782]

Item Overload [▶ 1785]

TwinCAT.TypeSystem Namespace [▶ 1184]

RpcMethodCollection.Item Property (String)

Gets the IRpcMethod [▶ 1597] with the specified method name.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IRpcMethod this[
    string methodName
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    methodName As String
) As IRpcMethod
    Get
```

Parameters

methodName	Type: System.String Name of the method.

Return Value

Type: IRpcMethod [▶ 1597]
RpcMethod.

Exceptions

Exception	Condition
KeyNotFoundException	

Reference

RpcMethodCollection Class [▶ 1782]

Item Overload [▶ 1785]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.2 RpcMethodCollection Methods

The RpcMethodCollection [▶ 1782] type exposes the following members.

Methods

	Name	Description
	Add [▶ 1788]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1789]	Gets a read only collection of this RpcMethodCollection [▶ 1782]
	Clear [▶ 1789]	Removes all items from the ICollection.T..
	Contains(String) [▶ 1790]	Determines whether this collection contains the specified method name.
	Contains(IRpcMethod) [▶ 1790]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1791]	Copies to.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1792]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1792]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1793]	Inserts an item to the IList.T. at the specified index.

	Name	Description
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1794]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1795]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetMethod(Int32, IRpcMethod.) [▶ 1796]	Tries to get the specified method.
	TryGetMethod(String, IRpcMethod.) [▶ 1796]	Tries to get the specified method.

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.2.1 RpcMethodCollection.Add Method

Adds an item to the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Add(
    IRpcMethod item
)
```

VB

```
Public Sub Add (
    item As IRpcMethod
)
```

Parameters

item	Type: TwinCAT.TypeSystem.IRpcMethod [▶ 1597] The object to add to the ICollection.T..

Implements

ICollection.T..Add(T)

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.2.2 RpcMethodCollection.AsReadOnly Method

Gets a read only collection of this RpcMethodCollection [▶ 1782]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyRpcMethodCollection AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlyRpcMethodCollection
```

Field Value

Type: ReadOnlyRpcMethodCollection [▶ 1762]
Returns a read only version of this RpcMethodCollection [▶ 1782]

Return Value

Type: ReadOnlyRpcMethodCollection [▶ 1762]
ReadOnlyRpcMethodCollection.

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.2.3 RpcMethodCollection.Clear Method

Removes all items from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Clear()
```

VB

```
Public Sub Clear
```

Implements

ICollection.T..Clear.

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.2.4 RpcMethodCollection.Contains Method

Overload List

	Name	Description
	Contains(String) [▶ 1790]	Determines whether this collection contains the specified method name.
	Contains(IRpcMethod) [▶ 1790]	Determines whether the ICollection.T. contains a specific value.

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

RpcMethodCollection.Contains Method (String)

Determines whether this collection contains the specified method name.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    string methodName
)
```

VB

```
Public Function Contains (
    methodName As String
) As Boolean
```

Parameters

methodName	Type: System.String Name of the method.

Return Value

Type: Boolean
true if contained.; otherwise, false.

Reference

RpcMethodCollection Class [▶ 1782]

Contains Overload [▶ 1790]

TwinCAT.TypeSystem Namespace [▶ 1184]

RpcMethodCollection.Contains Method (IRpcMethod)

Determines whether the ICollection.T. contains a specific value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    IRpcMethod item
)
```

VB

```
Public Function Contains (
    item As IRpcMethod
) As Boolean
```

Parameters

item	Type: TwinCAT.TypeSystem.IRpcMethod [▶ 1597] The object to locate in the ICollection.T..

Return Value

Type: Boolean
true if item is found in the ICollection.T.; otherwise, false.

Implements

ICollection.T..Contains(T)

Reference

RpcMethodCollection Class [▶ 1782]

Contains Overload [▶ 1790]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.2.5 RpcMethodCollection.CopyTo Method

Copies to.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void CopyTo(
    IRpcMethod[] array,
    int arrayIndex
)
```

VB

```
Public Sub CopyTo (
    array As IRpcMethod(),
    arrayIndex As Integer
)
```

Parameters

array	Type: .TwinCAT.TypeSystem.IRpcMethod [▶ 1597]. The array.
arrayIndex	Type: System.Int32 Index of the array.

Implements

ICollection.T..CopyTo(.T., Int32)

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.2.6 RpcMethodCollection.GetEnumerator Method

Returns an enumerator that iterates through the collection.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumerator<IRpcMethod> GetEnumerator()
```

VB

```
Public Function GetEnumerator As IEnumerator(Of IRpcMethod)
```

Return Value

Type: IEnumerator.IRpcMethod [▶ 1597].
A IEnumerator.T. that can be used to iterate through the collection.

Implements

IEnumerable.T..GetEnumerator.

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.2.7 RpcMethodCollection.IndexOf Method

Determines the index of a specific item in the IList.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int IndexOf(
    IRpcMethod item
)
```

VB

```
Public Function IndexOf (
    item As IRpcMethod
) As Integer
```

Parameters

item	Type: TwinCAT.TypeSystem.IRpcMethod [▶ 1597] The object to locate in the IList.T..

Return Value

Type: Int32
The index of item if found in the list; otherwise, -1.

Implements

IList.T..IndexOf(T)

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.2.8 RpcMethodCollection.Insert Method

Inserts an item to the IList.T. at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Insert(
    int index,
    IRpcMethod item
)
```

VB

```
Public Sub Insert (
    index As Integer,
    item As IRpcMethod
)
```

Parameters

index	Type: System.Int32 The zero-based index at which item should be inserted.

item	Type: TwinCAT.TypeSystem.IRpcMethod [▶ 1597] The object to insert into the IList.T..

Implements

IList.T..Insert(Int32, T)

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.2.9 RpcMethodCollection.Remove Method

Removes the first occurrence of a specific object from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Remove(
    IRpcMethod item
)
```

VB

```
Public Function Remove (
    item As IRpcMethod
) As Boolean
```

Parameters

item	Type: TwinCAT.TypeSystem.IRpcMethod [▶ 1597] The object to remove from the ICollection.T..

Return Value

Type: Boolean
true if item was successfully removed from the ICollection.T.; otherwise, false. This method also returns false if item is not found in the original ICollection.T..

Implements

ICollection.T..Remove(T)

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.2.10 RpcMethodCollection.RemoveAt Method

Removes the IList.T. item at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void RemoveAt(
    int index
)
```

VB

```
Public Sub RemoveAt (
    index As Integer
)
```

Parameters

index	Type: System.Int32 The zero-based index of the item to remove.

Implements

IList.T..RemoveAt(Int32)

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.96.2.11 RpcMethodCollection.TryGetMethod Method

Overload List

	Name	Description
	TryGetMethod(Int32, IRpcMethod.) [▶ 1796]	Tries to get the specified method.
	TryGetMethod(String, IRpcMethod.) [▶ 1796]	Tries to get the specified method.

Reference

RpcMethodCollection Class [▶ 1782]

TwinCAT.TypeSystem Namespace [▶ 1184]

RpcMethodCollection.TryGetMethod Method (Int32, IRpcMethod.)

Tries to get the specified method.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetMethod(
    int vTableIndex,
    out IRpcMethod method
)
```

VB

```
Public Function TryGetMethod (
    vTableIndex As Integer,
    <OutAttribute> ByRef method As IRpcMethod
) As Boolean
```

Parameters

vTableIndex	Type: System.Int32 VTable index.
method	Type: TwinCAT.TypeSystem.IRpcMethod [▶ 1597]. The method if fund, NULL otherwise.

Return Value

Type: Boolean
true if found, false otherwise.

Reference

RpcMethodCollection Class [▶ 1782]

TryGetMethod Overload [▶ 1795]

TwinCAT.TypeSystem Namespace [▶ 1184]

RpcMethodCollection.TryGetMethod Method (String, IRpcMethod.)

Tries to get the specified method.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetMethod(
    string methodName,
    out IRpcMethod method
)
```

VB

```
Public Function TryGetMethod (
    methodName As String,
    <OutAttribute> ByRef method As IRpcMethod
) As Boolean
```

Parameters

methodName	Type: System.String Name of the method.
method	Type: TwinCAT.TypeSystem.IRpcMethod [▶ 1597]. The method if fund, NULL otherwise.

Return Value

Type: Boolean
true if found, false otherwise.

Reference

RpcMethodCollection Class [▶ 1782]

TryGetMethod Overload [▶ 1795]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97 RpcMethodParameterCollection Class

Collection of RPC method parameters

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.RpcMethodParameterCollection
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class RpcMethodParameterCollection : IList<IRpcMethodParameter>,
    ICollection<IRpcMethodParameter>, IEnumerable<IRpcMethodParameter>, IEnumerable
```

VB

```
Public Class RpcMethodParameterCollection
    Implements IList(Of IRpcMethodParameter), ICollection(Of IRpcMethodParameter),
    IEnumerable(Of IRpcMethodParameter), IEnumerable
```

The RpcMethodParameterCollection type exposes the following members.

Properties

	Name	Description
	Count [▶ 1799]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1799]	Gets a value indicating whether the ICollection.T. is read-only.
	Item [▶ 1800]	Gets or sets the element at the specified index.

Methods

	Name	Description
	Add [▶ 1801]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1802]	Returns a read only version of this RpcMethodParameterCollection
	Clear [▶ 1802]	Removes all items from the ICollection.T..
	Contains [▶ 1803]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1804]	Copies to.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1804]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1805]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1805]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1806]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1807]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.1 RpcMethodParameterCollection Properties

The RpcMethodParameterCollection [▶ 1797] type exposes the following members.

Properties

	Name	Description
	Count [▶ 1799]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1799]	Gets a value indicating whether the ICollection.T. is read-only.
	Item [▶ 1800]	Gets or sets the element at the specified index.

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.1.1 RpcMethodParameterCollection.Count Property

Gets the number of elements contained in the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Count { get; }
```

VB

```
Public ReadOnly Property Count As Integer
    Get
```

Property Value

Type: Int32
The count.

Implements

ICollection.T..Count

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.1.2 RpcMethodParameterCollection.IsReadOnly Property

Gets a value indicating whether the ICollection.T. is read-only.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is read only; otherwise, false.

Implements

ICollection.T..IsReadOnly

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.1.3 RpcMethodParameterCollection.Item Property

Gets or sets the element at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IRpcMethodParameter this[
    int index
] { get; set; }
```

VB

```
Public Default Property Item (
    index As Integer
) As IRpcMethodParameter
    Get
    Set
```

Parameters

index	Type: System.Int32 The index.

Return Value

Type: IRpcMethodParameter [▶ 1601]
RpcMethodParameter.

Implements

IList.T..Item.Int32.

Exceptions

Exception	Condition
NotImplementedException	

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.2 RpcMethodParameterCollection Methods

The RpcMethodParameterCollection [▶ 1797] type exposes the following members.

Methods

	Name	Description
	Add [▶ 1801]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1802]	Returns a read only version of this RpcMethodParameterCollection [▶ 1797]
	Clear [▶ 1802]	Removes all items from the ICollection.T..
	Contains [▶ 1803]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1804]	Copies to.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1804]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1805]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1805]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1806]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1807]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.2.1 RpcMethodParameterCollection.Add Method

Adds an item to the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Add(
    IRpcMethodParameter item
)
```

VB

```
Public Sub Add (
    item As IRpcMethodParameter
)
```

Parameters

item	Type: TwinCAT.TypeSystem.IRpcMethodParameter [▶ 1601] The object to add to the ICollection.T..

Implements

ICollection.T..Add(T)

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.2.2 RpcMethodParameterCollection.AsReadOnly Method

Returns a read only version of this RpcMethodParameterCollection [▶ 1797]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyMethodParameterCollection AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlyMethodParameterCollection
```

Field Value

Type: ReadOnlyMethodParameterCollection [▶ 1759]
Collection as read only version.

Return Value

Type: ReadOnlyMethodParameterCollection [▶ 1759]
ReadOnlyMethodParameterCollection.

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.2.3 RpcMethodParameterCollection.Clear Method

Removes all items from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Clear()
```

VB

```
Public Sub Clear
```

Implements

ICollection.T..Clear.

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.2.4 RpcMethodParameterCollection.Contains Method

Determines whether the ICollection.T. contains a specific value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    IRpcMethodParameter item
)
```

VB

```
Public Function Contains (
    item As IRpcMethodParameter
) As Boolean
```

Parameters

item	Type: TwinCAT.TypeSystem.IRpcMethodParameter [▶ 1601] The object to locate in the ICollection.T..

Return Value

Type: Boolean
true if item is found in the ICollection.T.; otherwise, false.

Implements

ICollection.T..Contains(T)

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.2.5 RpcMethodParameterCollection.CopyTo Method

Copies to.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void CopyTo(
    IRpcMethodParameter[] array,
    int arrayIndex
)
```

VB

```
Public Sub CopyTo (
    array As IRpcMethodParameter(),
    arrayIndex As Integer
)
```

Parameters

array	Type: .TwinCAT.TypeSystem.IRpcMethodParameter [▶ 1601]. The array.
arrayIndex	Type: System.Int32 Index of the array.

Implements

ICollection.T..CopyTo(.T., Int32)

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.2.6 RpcMethodParameterCollection.GetEnumerator Method

Returns an enumerator that iterates through the collection.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumerator<IRpcMethodParameter> GetEnumerator()
```

VB

```
Public Function GetEnumerator As IEnumerator(Of IRpcMethodParameter)
```

Return Value

Type: IEnumerator.IRpcMethodParameter [▶ 1601].
A IEnumerator.T. that can be used to iterate through the collection.

Implements

IEnumerable.T..GetEnumerator.

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.2.7 RpcMethodParameterCollection.IndexOf Method

Determines the index of a specific item in the IList.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int IndexOf(
    IRpcMethodParameter item
)
```

VB

```
Public Function IndexOf (
    item As IRpcMethodParameter
) As Integer
```

Parameters

item	Type: TwinCAT.TypeSystem.IRpcMethodParameter [▶ 1601] The object to locate in the IList.T..

Return Value

Type: Int32
The index of item if found in the list; otherwise, -1.

Implements

IList.T..IndexOf(T)

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.2.8 RpcMethodParameterCollection.Insert Method

Inserts an item to the IList.T. at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Insert(
    int index,
    IRpcMethodParameter item
)
```

VB

```
Public Sub Insert (
    index As Integer,
    item As IRpcMethodParameter
)
```

Parameters

index	Type: System.Int32 The zero-based index at which item should be inserted.
item	Type: TwinCAT.TypeSystem.IRpcMethodParameter [▶ 1601] The object to insert into the IList.T..

Implements

IList.T..Insert(Int32, T)

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.2.9 RpcMethodParameterCollection.Remove Method

Removes the first occurrence of a specific object from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Remove(
    IRpcMethodParameter item
)
```

VB

```
Public Function Remove (
    item As IRpcMethodParameter
) As Boolean
```

Parameters

item	Type: TwinCAT.TypeSystem.IRpcMethodParameter [▶ 1601] The object to remove from the ICollection.T..

Return Value

Type: Boolean
true if item was successfully removed from the ICollection.T.; otherwise, false. This method also returns false if item is not found in the original ICollection.T..

Implements

ICollection.T..Remove(T)

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.97.2.10 RpcMethodParameterCollection.RemoveAt Method

Removes the IList.T. item at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void RemoveAt(
    int index
)
```

VB

```
Public Sub RemoveAt (
    index As Integer
)
```

Parameters

index	Type: System.Int32 The zero-based index of the item to remove.

Implements

IList.T..RemoveAt(Int32)

Reference

RpcMethodParameterCollection Class [▶ 1797]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98 SubItemCollection Class

Class SubItemCollection.

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.SubItemCollection
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SubItemCollection : IList<ITcAdsSubItem>,
    ICollection<ITcAdsSubItem>, IEnumerable<ITcAdsSubItem>, IEnumerable
```

VB

```
Public Class SubItemCollection
    Implements IList(Of ITcAdsSubItem), ICollection(Of ITcAdsSubItem),
    IEnumerable(Of ITcAdsSubItem), IEnumerable
```

The SubItemCollection type exposes the following members.

Properties

	Name	Description
	Count [▶ 1809]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1810]	Gets a value indicating whether the ICollection.T. is read-only.
	Item [▶ 1810]	Gets or sets the element at the specified index.

Methods

	Name	Description
	Add [▶ 1812]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1812]	Gets a read only version of this SubItemCollection.
	Clear [▶ 1813]	Removes all items from the ICollection.T..
	Contains [▶ 1813]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1814]	Copies to.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1815]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1815]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1816]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)

	Name	Description
	Remove [▶ 1817]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1817]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.1 SubItemCollection Properties

The SubItemCollection [▶ 1807] type exposes the following members.

Properties

	Name	Description
	Count [▶ 1809]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1810]	Gets a value indicating whether the ICollection.T. is read-only.
	Item [▶ 1810]	Gets or sets the element at the specified index.

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.1.1 SubItemCollection.Count Property

Gets the number of elements contained in the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Count { get; }
```

VB

```
Public ReadOnly Property Count As Integer
    Get
```

Property Value

Type: Int32
The count.

Implements

ICollection.T..Count

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.1.2 SubItemCollection.IsReadOnly Property

Gets a value indicating whether the ICollection.T. is read-only.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is read only; otherwise, false.

Implements

ICollection.T..IsReadOnly

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.1.3 SubItemCollection.Item Property

Gets or sets the element at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ITcAdsSubItem this[
    int index
] { get; set; }
```

VB

```
Public Default Property Item (
    index As Integer
) As ITcAdsSubItem
    Get
    Set
```

Parameters

index	Type: System.Int32 The index.

Return Value

Type: ITcAdsSubItem [▶ 544]
ITcAdsSubItem.

Implements

IList.T..Item.Int32.

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.2 SubItemCollection Methods

The SubItemCollection [▶ 1807] type exposes the following members.

Methods

	Name	Description
	Add [▶ 1812]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1812]	Gets a read only version of this SubItemCollection [▶ 1807].
	Clear [▶ 1813]	Removes all items from the ICollection.T..
	Contains [▶ 1813]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1814]	Copies to.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1815]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1815]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1816]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1817]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1817]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.2.1 SubItemCollection.Add Method

Adds an item to the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Add(
    ITcAdsSubItem item
)
```

VB

```
Public Sub Add (
    item As ITcAdsSubItem
)
```

Parameters

item	Type: TwinCAT.Ads.ITcAdsSubItem [▶ 544] The object to add to the ICollection.T..

Implements

ICollection.T..Add(T)

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.2.2 SubItemCollection.AsReadOnly Method

Gets a read only version of this SubItemCollection [▶ 1807].

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlySubItemCollection AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlySubItemCollection
```

Field Value

Type: ReadOnlySubItemCollection [▶ 1768]
As read only.

Return Value

Type: ReadOnlySubItemCollection [▶ 1768]
ReadOnlySubItemCollection.

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.2.3 SubItemCollection.Clear Method

Removes all items from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Clear()
```

VB

```
Public Sub Clear
```

Implements

ICollection.T..Clear.

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.2.4 SubItemCollection.Contains Method

Determines whether the ICollection.T. contains a specific value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    ITcAdsSubItem item
)
```

VB

```
Public Function Contains (
    item As ITcAdsSubItem
) As Boolean
```

Parameters

item	Type: TwinCAT.Ads.ITcAdsSubItem [▶ 544] The object to locate in the ICollection.T..

Return Value

Type: Boolean
true if item is found in the ICollection.T.; otherwise, false.

Implements

ICollection.T..Contains(T)

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.2.5 SubItemCollection.CopyTo Method

Copies to.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void CopyTo(
    ITcAdsSubItem[] array,
    int arrayIndex
)
```

VB

```
Public Sub CopyTo (
    array As ITcAdsSubItem(),
    arrayIndex As Integer
)
```

Parameters

array	Type: .TwinCAT.Ads.ITcAdsSubItem [▶ 544]. The array.
arrayIndex	Type: System.Int32 Index of the array.

Implements

ICollection.T..CopyTo(.T., Int32)

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.2.6 SubItemCollection.GetEnumerator Method

Returns an enumerator that iterates through the collection.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumerator<ITcAdsSubItem> GetEnumerator()
```

VB

```
Public Function GetEnumerator As IEnumerator(Of ITcAdsSubItem)
```

Return Value

Type: IEnumerator.ITcAdsSubItem [▶ 544].
A IEnumerator.T. that can be used to iterate through the collection.

Implements

IEnumerable.T..GetEnumerator.

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.2.7 SubItemCollection.IndexOf Method

Determines the index of a specific item in the IList.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int IndexOf(
    ITcAdsSubItem item
)
```

VB

```
Public Function IndexOf (
    item As ITcAdsSubItem
) As Integer
```

Parameters

item	Type: TwinCAT.Ads.ITcAdsSubItem [▶ 544] The object to locate in the IList.T..

Return Value

Type: Int32
The index of item if found in the list; otherwise, -1.

Implements

IList.T..IndexOf(T)

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.2.8 SubItemCollection.Insert Method

Inserts an item to the IList.T. at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Insert(
    int index,
    ITcAdsSubItem item
)
```

VB

```
Public Sub Insert (
    index As Integer,
    item As ITcAdsSubItem
)
```

Parameters

index	Type: System.Int32 The zero-based index at which item should be inserted.
item	Type: TwinCAT.Ads.ITcAdsSubItem [▶ 544] The object to insert into the IList.T..

Implements

IList.T..Insert(Int32, T)

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.2.9 SubItemCollection.Remove Method

Removes the first occurrence of a specific object from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Remove(
    ITcAdsSubItem item
)
```

VB

```
Public Function Remove (
    item As ITcAdsSubItem
) As Boolean
```

Parameters

item	Type: TwinCAT.Ads.ITcAdsSubItem [▶ 544] The object to remove from the ICollection.T..

Return Value

Type: Boolean
true if item was successfully removed from the ICollection.T.; otherwise, false. This method also returns false if item is not found in the original ICollection.T..

Implements

ICollection.T..Remove(T)

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.98.2.10 SubItemCollection.RemoveAt Method

Removes the IList.T. item at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void RemoveAt(
    int index
)
```

VB

```
Public Sub RemoveAt (
    index As Integer
)
```

Parameters

index	Type: System.Int32 The zero-based index of the item to remove.

Implements

IList.T..RemoveAt(Int32)

Reference

SubItemCollection Class [▶ 1807]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.99 SymbolAccessRights Enumeration

Enum specifying Access Rights to symbols

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[FlagsAttribute]
public enum SymbolAccessRights
```

VB

```
<FlagsAttribute>
Public Enumeration SymbolAccessRights
```

Members

	Member name	Value	Description
	None	0	None / Uninitialized
	Read	1	Read-Access
	Write	2	Write-Access
	MethodInvoke	4	Right to Invoke Methods / RPC Invoke
	ReadWrite	3	Read / Write Access
	All	7	Full Access

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.100 SymbolCollection Class

Interface represents a collection of ISymbol [▶ 1634] objects.

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.Generic.InstanceCollection [▶ 1887].ISymbol [▶ 1634].
    TwinCAT.TypeSystem.Generic.SymbolCollection [▶ 1964].ISymbol [▶ 1634].
      TwinCAT.TypeSystem.SymbolCollection

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SymbolCollection : SymbolCollection<ISymbol>,
    ISymbolCollection, ISymbolCollection<ISymbol>, IInstanceCollection<ISymbol>,
    IList<ISymbol>, ICollection<ISymbol>, IEnumerable<ISymbol>,
    IEnumerable
```

VB

```
Public Class SymbolCollection
    Inherits SymbolCollection(Of ISymbol)
    Implements ISymbolCollection, ISymbolCollection(Of ISymbol),
    IInstanceCollection(Of ISymbol), IList(Of ISymbol), ICollection(Of ISymbol),
    IEnumerable(Of ISymbol), IEnumerable
```

The SymbolCollection type exposes the following members.

Properties

	Name	Description
	Count [▶ 1891]	Gets the collection count. (Inherited from InstanceCollection.T. [▶ 1887].)
	IsReadOnly [▶ 1892]	Gets a value indicating whether this instance is read only. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.Int32. [▶ 1893]	Gets or sets the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.String. [▶ 1894]	Gets the IInstance [▶ 1556] with the specified instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	Mode [▶ 1894]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from InstanceCollection.T. [▶ 1887].)

Methods

	Name	Description
	Add [▶ 1896]	Adds the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	AddRange [▶ 1897]	Adds the specified items to this collection. (Inherited from InstanceCollection.T. [▶ 1887].)
	AsReadOnly [▶ 1822]	Returns a Read only version of this collection (shallow copy).
	Clear [▶ 1898]	Clears this instance. (Inherited from InstanceCollection.T. [▶ 1887].)
	Clone [▶ 1823]	Clones this instance.
	Contains(String) [▶ 1899]	Determines whether this collection contains an IInstance [▶ 1556] with the specified InstanceName / InstancePath (Inherited from InstanceCollection.T. [▶ 1887].)
	Contains(T) [▶ 1899]	Determines whether this collection contains the specified IInstance [▶ 1556] (Inherited from InstanceCollection.T. [▶ 1887].)

	Name	Description
	ContainsName [▶ 1900]	Determines whether the specified instance name contains name. (Inherited from InstanceCollection.T. [▶ 1887].)
	CopyTo [▶ 1901]	Copies this InstanceCollection.T. [▶ 1887] to the specified array. (Inherited from InstanceCollection.T. [▶ 1887].)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1902]	Gets the enumerator. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1902]	Gets the IInstance [▶ 1556]by instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetInstanceByName [▶ 1903]	Gets the name of the instance by. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1904]	Determines the index of the specified IInstance [▶ 1556]. (Inherited from InstanceCollection.T. [▶ 1887].)
	Insert [▶ 1905]	Inserts the specified IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1905]	Removes the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	RemoveAt [▶ 1906]	Removes the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1907]	Tries to get the IInstance [▶ 1556]. of the specified path. (Inherited from InstanceCollection.T. [▶ 1887].)
	TryGetInstanceByName [▶ 1908]	Tries to get Instnaces by name. (Inherited from InstanceCollection.T. [▶ 1887].)
	TryGetInstances [▶ 1969]	Try to get instances with predicate function (Inherited from SymbolCollection.T. [▶ 1964].)

Fields

	Name	Description
	_list [▶ 1909]	The _list (Inherited from InstanceCollection.T. [▶ 1887].)
	_pathDict [▶ 1909]	The _path dictionary (Inherited from InstanceCollection.T. [▶ 1887].)
	mode [▶ 1910]	The mode this InstanceCollection.T. [▶ 1887] is working in. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.100.1 SymbolCollection Properties

The SymbolCollection [▶ 1818] type exposes the following members.

Properties

	Name	Description
	Count [▶ 1891]	Gets the collection count. (Inherited from InstanceCollection.T. [▶ 1887].)
	IsReadOnly [▶ 1892]	Gets a value indicating whether this instance is read only. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.Int32. [▶ 1893]	Gets or sets the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.String. [▶ 1894]	Gets the IInstance [▶ 1556] with the specified instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	Mode [▶ 1894]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

SymbolCollection Class [▶ 1818]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.100.2 SymbolCollection Methods

The SymbolCollection [▶ 1818] type exposes the following members.

Methods

	Name	Description
	Add [▶ 1896]	Adds the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	AddRange [▶ 1897]	Adds the specified items to this collection. (Inherited from InstanceCollection.T. [▶ 1887].)
	AsReadOnly [▶ 1822]	Returns a Read only version of this collection (shallow copy).
	Clear [▶ 1898]	Clears this instance. (Inherited from InstanceCollection.T. [▶ 1887].)
	Clone [▶ 1823]	Clones this instance.
	Contains(String) [▶ 1899]	Determines whether this collection contains an IInstance [▶ 1556] with the specified InstanceName / InstancePath (Inherited from InstanceCollection.T. [▶ 1887].)
	Contains(T) [▶ 1899]	Determines whether this collection contains the specified IInstance [▶ 1556] (Inherited from InstanceCollection.T. [▶ 1887].)
	ContainsName [▶ 1900]	Determines whether the specified instance name contains name. (Inherited from InstanceCollection.T. [▶ 1887].)
	CopyTo [▶ 1901]	Copies this InstanceCollection.T. [▶ 1887] to the specified array. (Inherited from InstanceCollection.T. [▶ 1887].)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	GetEnumerator [▶ 1902]	Gets the enumerator. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1902]	Gets the IInstance [▶ 1556]by instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetInstanceByName [▶ 1903]	Gets the name of the instance by. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1904]	Determines the index of the specified IInstance [▶ 1556]. (Inherited from InstanceCollection.T. [▶ 1887].)
	Insert [▶ 1905]	Inserts the specified IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1905]	Removes the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	RemoveAt [▶ 1906]	Removes the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1907]	Tries to get the IInstance [▶ 1556]. of the specified path. (Inherited from InstanceCollection.T. [▶ 1887].)
	TryGetInstanceByName [▶ 1908]	Tries to get Instnaces by name. (Inherited from InstanceCollection.T. [▶ 1887].)
	TryGetInstances [▶ 1969]	Try to get instances with predicate function (Inherited from SymbolCollection.T. [▶ 1964].)

Reference

SymbolCollection Class [▶ 1818]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.100.2.1 SymbolCollection.AsReadOnly Method

Returns a Read only version of this collection (shallow copy).

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlySymbolCollection AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlySymbolCollection
```

Return Value

Type: ReadOnlySymbolCollection [▶ 1772]
Read only collection.

Reference

SymbolCollection Class [▶ 1818]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.100.2.2 SymbolCollection.Clone Method

Clones this instance.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolCollection Clone()
```

VB

```
Public Function Clone As SymbolCollection
```

Return Value

Type: SymbolCollection [▶ 1818]
Cloned SymbolCollection [▶ 1818].

Reference

SymbolCollection Class [▶ 1818]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.100.3 SymbolCollection Fields

The SymbolCollection [▶ 1818] type exposes the following members.

Fields

	Name	Description
	_list [▶ 1909]	The _list (Inherited from InstanceCollection.T. [▶ 1887].)
	_pathDict [▶ 1909]	The _path dictionary (Inherited from InstanceCollection.T. [▶ 1887].)
	mode [▶ 1910]	The mode this InstanceCollection.T. [▶ 1887] is working in. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

SymbolCollection Class [▶ 1818]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.101 TypeAttribute Class

ADS Attribute

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.TypeAttribute
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class TypeAttribute : ITypeAttribute
```

VB

```
Public Class TypeAttribute
    Implements ITypeAttribute
```

The TypeAttribute type exposes the following members.

Properties

	Name	Description
	Name [▶ 1825]	Name of the Attribute
	Value [▶ 1826]	Gets the value of the attribute

Methods

	Name	Description
	Equals [▶ 1827]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 1827]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Operators

	Name	Description
	Equality [▶ 1828]	Operator==
	Inequality [▶ 1829]	Implements the != operator.

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.101.1 TypeAttribute Properties

The TypeAttribute [▶ 1824] type exposes the following members.

Properties

	Name	Description
	Name [▶ 1825]	Name of the Attribute
	Value [▶ 1826]	Gets the value of the attribute

Reference

TypeAttribute Class [▶ 1824]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.101.1.1 TypeAttribute.Name Property

Name of the Attribute

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Name { get; }
```

VB

```
Public ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Implements

ITypeAttribute.Name [▶ 1652]

Reference

TypeAttribute Class [▶ 1824]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.101.1.2 TypeAttribute.Value Property

Gets the value of the attribute

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string Value { get; }
```

VB

```
Public ReadOnly Property Value As String
    Get
```

Property Value

Type: String
The value.

Implements

ITypeAttribute.Value [▶ 1653]

Reference

TypeAttribute Class [▶ 1824]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.101.2 TypeAttribute Methods

The TypeAttribute [▶ 1824] type exposes the following members.

Methods

	Name	Description
	Equals [▶ 1827]	Equals (Overrides Object.Equals(Object).)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode [▶ 1827]	Gets the HashCode of the Address (Overrides Object.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

TypeAttribute Class [▶ 1824]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.101.2.1 TypeAttribute.Equals Method

Equals

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override bool Equals(
    Object obj
)
```

VB

```
Public Overrides Function Equals (
    obj As Object
) As Boolean
```

Parameters

obj	Type: System.Object The object to compare with the current object.

Return Value

Type: Boolean
true if the specified Object is equal to this instance; otherwise, false.

Reference

TypeAttribute Class [▶ 1824]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.101.2.2 TypeAttribute.GetHashCode Method

Gets the HashCode of the Address

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public override int GetHashCode()
```

VB

```
Public Overrides Function GetHashCode As Integer
```

Return Value

Type: Int32
A hash code for this instance, suitable for use in hashing algorithms and data structures like a hash table.

Reference

TypeAttribute Class [▶ 1824]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.101.3 TypeAttribute Operators

The TypeAttribute [▶ 1824] type exposes the following members.

Operators

	Name	Description
	Equality [▶ 1828]	Operator==
	Inequality [▶ 1829]	Implements the != operator.

Reference

TypeAttribute Class [▶ 1824]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.101.3.1 TypeAttribute.Equality Operator

Operator==

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator ==(
    TypeAttribute o1,
    TypeAttribute o2
)
```

VB

```
Public Shared Operator = (
    o1 As TypeAttribute,
    o2 As TypeAttribute
) As Boolean
```

Parameters

o1	Type: TwinCAT.TypeSystem.TypeAttribute [▶ 1824] The o1.
o2	Type: TwinCAT.TypeSystem.TypeAttribute [▶ 1824] The o2.

Return Value

Type: Boolean
The result of the operator.

Reference

TypeAttribute Class [▶ 1824]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.101.3.2 TypeAttribute.Inequality Operator

Implements the != operator.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public static bool operator !=(
    TypeAttribute o1,
    TypeAttribute o2
)
```

VB

```
Public Shared Operator <> (
    o1 As TypeAttribute,
    o2 As TypeAttribute
) As Boolean
```

Parameters

o1	Type: TwinCAT.TypeSystem.TypeAttribute [▶ 1824] The o1.
o2	Type: TwinCAT.TypeSystem.TypeAttribute [▶ 1824] The o2.

Return Value

Type: Boolean
The result of the operator.

Reference

TypeAttribute Class [▶ 1824]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102 TypeAttributeCollection Class

Collection of AdsAttributes [▶ 1651]

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.TypeAttributeCollection
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class TypeAttributeCollection : IList<ITypeAttribute>,
    ICollection<ITypeAttribute>, IEnumerable<ITypeAttribute>, IEnumerable
```

VB

```
Public Class TypeAttributeCollection
    Implements IList(Of ITypeAttribute), ICollection(Of ITypeAttribute),
    IEnumerable(Of ITypeAttribute), IEnumerable
```

The TypeAttributeCollection type exposes the following members.

Constructors

	Name	Description
	TypeAttributeCollection. [▶ 1832]	Initializes a new instance of the TypeAttributeCollection class.
	TypeAttributeCollection(IEnumerable.ITypeAttribute.) [▶ 1832]	Initializes a new instance of the TypeAttributeCollection class.

Properties

	Name	Description
	Count [▶ 1833]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1833]	Gets a value indicating whether the ICollection.T. is read-only.
	Item.Int32. [▶ 1834]	Gets or sets the element at the specified index.
	Item.String. [▶ 1835]	Gets the String with the specified name.

Methods

	Name	Description
	Add [▶ 1837]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1837]	Gets a read only version of this TypeAttributeCollection
	Clear [▶ 1838]	Removes all items from the ICollection.T..
	Contains(String) [▶ 1839]	Determines whether this TypeAttributeCollection contains the ITypeAttribute [▶ 1651] with the specified name.
	Contains(ITypeAttribute) [▶ 1840]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1840]	Copies to.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	GetEnumerator [▶ 1841]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1842]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1842]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove(String) [▶ 1843]	Removes the specified ITypeAttribute [▶ 1651] from the TypeAttributeCollection
	Remove(ITypeAttribute) [▶ 1844]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1845]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetAttribute [▶ 1845]	Tries to get the specified ITypeAttribute [▶ 1651]
	TryGetValue [▶ 1846]	Tries to get the specified Attribute value.

Fields

	Name	Description
	list [▶ 1847]	List of Attributes

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.1 TypeAttributeCollection Constructor

Overload List

	Name	Description
	TypeAttributeCollection. [▶ 1832]	Initializes a new instance of the TypeAttributeCollection [▶ 1829] class.
	TypeAttributeCollection(IEnumerable.ITypeAttribute.) [▶ 1832]	Initializes a new instance of the TypeAttributeCollection [▶ 1829] class.

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.1.1 TypeAttributeCollection Constructor

Initializes a new instance of the TypeAttributeCollection [▶ 1829] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TypeAttributeCollection()
```

VB

```
Public Sub New
```

Reference

TypeAttributeCollection Class [▶ 1829]

TypeAttributeCollection Overload [▶ 1831]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.1.2 TypeAttributeCollection Constructor (IEnumerable.ITypeAttribute.)

Initializes a new instance of the TypeAttributeCollection [▶ 1829] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public TypeAttributeCollection(
    IEnumerable<ITypeAttribute> coll
)
```

VB

```
Public Sub New (
    coll As IEnumerable(Of ITypeAttribute)
)
```

Parameters

coll	Type: System.Collections.Generic.IEnumerable.ITypeAttribute [▶ 1651]. The coll.

Reference

TypeAttributeCollection Class [▶ 1829]

TypeAttributeCollection Overload [▶ 1831]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.2 TypeAttributeCollection Properties

The TypeAttributeCollection [▶ 1829] type exposes the following members.

Properties

	Name	Description
	Count [▶ 1833]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1833]	Gets a value indicating whether the ICollection.T. is read-only.
	Item.Int32. [▶ 1834]	Gets or sets the element at the specified index.
	Item.String. [▶ 1835]	Gets the String with the specified name.

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.2.1 TypeAttributeCollection.Count Property

Gets the number of elements contained in the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Count { get; }
```

VB

```
Public ReadOnly Property Count As Integer
    Get
```

Property Value

Type: Int32
The count.

Implements

ICollection.T..Count

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.2.2 TypeAttributeCollection.IsReadOnly Property

Gets a value indicating whether the ICollection.T. is read-only.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is read only; otherwise, false.

Implements

ICollection.T..IsReadOnly

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.2.3 TypeAttributeCollection.Item Property

Overload List

	Name	Description
	Item.Int32. [▶ 1834]	Gets or sets the element at the specified index.
	Item.String. [▶ 1835]	Gets the String with the specified name.

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

TypeAttributeCollection.Item Property (Int32)

Gets or sets the element at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ITypeAttribute this[
    int index
] { get; set; }
```

VB

```
Public Default Property Item (
    index As Integer
) As ITypeAttribute
    Get
    Set
```

Parameters

index	Type: System.Int32 The index.

Return Value

Type: ITypeAttribute [▶ 1651]
AdsAttribute.

Implements

IList.T..Item.Int32.

Reference

TypeAttributeCollection Class [▶ 1829]

Item Overload [▶ 1834]

TwinCAT.TypeSystem Namespace [▶ 1184]

TypeAttributeCollection.Item Property (String)

Gets the String with the specified name.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public string this[
    string name
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    name As String
) As String
    Get
```

Parameters

name	Type: System.String The name.

Return Value

Type: String
System.String.

Exceptions

Exception	Condition
KeyNotFoundException	

Reference

TypeAttributeCollection Class [▶ 1829]

Item Overload [▶ 1834]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3 TypeAttributeCollection Methods

The TypeAttributeCollection [▶ 1829] type exposes the following members.

Methods

	Name	Description
	Add [▶ 1837]	Adds an item to the ICollection.T..
	AsReadOnly [▶ 1837]	Gets a read only version of this TypeAttributeCollection [▶ 1829]
	Clear [▶ 1838]	Removes all items from the ICollection.T..
	Contains(String) [▶ 1839]	Determines whether this TypeAttributeCollection [▶ 1829] contains the ITypeAttribute [▶ 1651] with the specified name.
	Contains(ITypeAttribute) [▶ 1840]	Determines whether the ICollection.T. contains a specific value.
	CopyTo [▶ 1840]	Copies to.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1841]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1842]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1842]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove(String) [▶ 1843]	Removes the specified ITypeAttribute [▶ 1651] from the TypeAttributeCollection [▶ 1829]
	Remove(ITypeAttribute) [▶ 1844]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1845]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)

	Name	Description
	TryGetAttribute [▶ 1845]	Tries to get the specified ITypeAttribute [▶ 1651]
	TryGetValue [▶ 1846]	Tries to get the specified Attribute value.

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3.1 TypeAttributeCollection.Add Method

Adds an item to the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Add(
    ITypeAttribute item
)
```

VB

```
Public Sub Add (
    item As ITypeAttribute
)
```

Parameters

item	Type: TwinCAT.TypeSystem.ITypeAttribute [▶ 1651] The object to add to the ICollection.T..

Implements

ICollection.T..Add(T)

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3.2 TypeAttributeCollection.AsReadOnly Method

Gets a read only version of this TypeAttributeCollection [▶ 1829]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyTypeAttributeCollection AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlyTypeAttributeCollection
```

Field Value

Type: ReadOnlyTypeAttributeCollection [▶ 1776]
As read only.

Return Value

Type: ReadOnlyTypeAttributeCollection [▶ 1776]
ReadOnlyAttributeCollection.

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3.3 TypeAttributeCollection.Clear Method

Removes all items from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Clear()
```

VB

```
Public Sub Clear
```

Implements

ICollection.T..Clear.

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3.4 TypeAttributeCollection.Contains Method

Overload List

	Name	Description
	Contains(String) [▶ 1839]	Determines whether this TypeAttributeCollection [▶ 1829] contains the ITypeAttribute [▶ 1651] with the specified name.
	Contains(ITypeAttribute) [▶ 1840]	Determines whether the ICollection.T. contains a specific value.

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

TypeAttributeCollection.Contains Method (String)

Determines whether this TypeAttributeCollection [▶ 1829] contains the ITypeAttribute [▶ 1651] with the specified name.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    string name
)
```

VB

```
Public Function Contains (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name.	

Return Value

Type: Boolean
true if [contains] [the specified name]; otherwise, false.

Reference

TypeAttributeCollection Class [▶ 1829]

Contains Overload [▶ 1839]

TwinCAT.TypeSystem Namespace [▶ 1184]

TypeAttributeCollection.Contains Method (ITypeAttribute)

Determines whether the ICollection.T. contains a specific value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    ITypeAttribute item
)
```

VB

```
Public Function Contains (
    item As ITypeAttribute
) As Boolean
```

Parameters

item	Type: TwinCAT.TypeSystem.ITypeAttribute [▶ 1651] The object to locate in the ICollection.T..

Return Value

Type: Boolean
true if item is found in the ICollection.T.; otherwise, false.

Implements

ICollection.T..Contains(T)

Reference

TypeAttributeCollection Class [▶ 1829]

Contains Overload [▶ 1839]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3.5 TypeAttributeCollection.CopyTo Method

Copies to.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void CopyTo(
    ITypeAttribute[] array,
    int arrayIndex
)
```

VB

```
Public Sub CopyTo (
    array As ITypeAttribute(),
    arrayIndex As Integer
)
```

Parameters

array	Type: .TwinCAT.TypeSystem.ITypeAttribute [▶ 1651]. The array.
arrayIndex	Type: System.Int32 Index of the array.

Implements

ICollection.T..CopyTo(.T., Int32)

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3.6 TypeAttributeCollection.GetEnumerator Method

Returns an enumerator that iterates through the collection.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumerator<ITypeAttribute> GetEnumerator()
```

VB

```
Public Function GetEnumerator As IEnumerator(Of ITypeAttribute)
```

Return Value

Type: IEnumerator.ITypeAttribute [▶ 1651].
A IEnumerator.T. that can be used to iterate through the collection.

Implements

IEnumerable.T..GetEnumerator.

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3.7 TypeAttributeCollection.IndexOf Method

Determines the index of a specific item in the IList.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int IndexOf(
    ITypeAttribute item
)
```

VB

```
Public Function IndexOf (
    item As ITypeAttribute
) As Integer
```

Parameters

item	Type: TwinCAT.TypeSystem.ITypeAttribute [▶ 1651] The object to locate in the IList.T..

Return Value

Type: Int32
The index of item if found in the list; otherwise, -1.

Implements

IList.T..IndexOf(T)

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3.8 TypeAttributeCollection.Insert Method

Inserts an item to the IList.T. at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Insert(
    int index,
    ITypeAttribute item
)
```

VB

```
Public Sub Insert (
    index As Integer,
    item As ITypeAttribute
)
```

Parameters

index	Type: System.Int32 The zero-based index at which item should be inserted.
item	Type: TwinCAT.TypeSystem.ITypeAttribute [▶ 1651] The object to insert into the IList.T..

Implements

IList.T..Insert(Int32, T)

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3.9 TypeAttributeCollection.Remove Method

Overload List

	Name	Description
	Remove(String) [▶ 1843]	Removes the specified ITypeAttribute [▶ 1651] from the TypeAttributeCollection [▶ 1829]
	Remove(ITypeAttribute) [▶ 1844]	Removes the first occurrence of a specific object from the ICollection.T..

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

TypeAttributeCollection.Remove Method (String)

Removes the specified ITypeAttribute [▶ 1651] from the TypeAttributeCollection [▶ 1829]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Remove(
    string name
)
```

VB

```
Public Function Remove (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

TypeAttributeCollection Class [▶ 1829]

Remove Overload [▶ 1843]

TwinCAT.TypeSystem Namespace [▶ 1184]

TypeAttributeCollection.Remove Method (ITypeAttribute)

Removes the first occurrence of a specific object from the ICollection.T..

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Remove(
    ITypeAttribute item
)
```

VB

```
Public Function Remove (
    item As ITypeAttribute
) As Boolean
```

Parameters

item	Type: TwinCAT.TypeSystem.ITypeAttribute [▶ 1651] The object to remove from the ICollection.T..

Return Value

Type: Boolean
true if item was successfully removed from the ICollection.T.; otherwise, false. This method also returns false if item is not found in the original ICollection.T..

Implements

ICollection.T..Remove(T)

Reference

TypeAttributeCollection Class [▶ 1829]

Remove Overload [▶ 1843]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3.10 TypeAttributeCollection.RemoveAt Method

Removes the IList.T. item at the specified index.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void RemoveAt(
    int index
)
```

VB

```
Public Sub RemoveAt (
    index As Integer
)
```

Parameters

index	Type: System.Int32 The zero-based index of the item to remove.

Implements

IList.T..RemoveAt(Int32)

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3.11 TypeAttributeCollection.TryGetAttribute Method

Tries to get the specified ITypeAttribute [▶ 1651]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetAttribute(
    string name,
    out ITypeAttribute att
)
```

VB

```
Public Function TryGetAttribute (
    name As String,
    <OutAttribute> ByRef att As ITypeAttribute
) As Boolean
```

Parameters

name	Type: System.String The name of the ITypeAttribute [▶ 1651].
att	Type: TwinCAT.TypeSystem.ITypeAttribute [▶ 1651]. The att.

Return Value

Type: Boolean
true if found, false otherwise.

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.3.12 TypeAttributeCollection.TryGetValue Method

Tries to get the specified Attribute value.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetValue(
    string name,
    out string value
)
```

VB

```
Public Function TryGetValue (
    name As String,
    <OutAttribute> ByRef value As String
) As Boolean
```

Parameters

name	Type: System.String The name.
value	Type: System.String. The value.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.4 TypeAttributeCollection Fields

The TypeAttributeCollection [▶ 1829] type exposes the following members.

Fields

	Name	Description
	list [▶ 1847]	List of Attributes

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.102.4.1 TypeAttributeCollection.list Field

List of Attributes

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected List<ITypeAttribute> list
```

VB

```
Protected list As List(Of ITypeAttribute)
```

Field Value

Type: List.ITypeAttribute [▶ 1651].

Reference

TypeAttributeCollection Class [▶ 1829]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.103 ValueChangedArgs Class

Event args for the ValueChanged [▶ 1694] event.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.TypeSystem.ValueChangedBaseArgs [▶ 1850]

TwinCAT.TypeSystem.ValueChangedArgs
Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ValueChangedArgs : ValueChangedBaseArgs
```

VB

```
Public Class ValueChangedArgs
    Inherits ValueChangedBaseArgs
```

The ValueChangedArgs type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Fields

	Name	Description
	Symbol [▶ 1852]	Symbol that caused the event. (Inherited from ValueChangedBaseArgs [▶ 1850].)
	UtcLocalSystemTime [▶ 1853]	Notification Time stamp of the local system (user/desktop time) in UTC (Inherited from ValueChangedBaseArgs [▶ 1850].)
	UtcRtime [▶ 1853]	Notification Time stamp of the Real time System in UTC (Inherited from ValueChangedBaseArgs [▶ 1850].)
	Value [▶ 1849]	The new received Value

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.103.1 ValueChangedArgs Methods

The ValueChangedArgs [▶ 1847] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

ValueChangedArgs Class [▶ 1847]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.103.2 ValueChangedArgs Fields

The ValueChangedArgs [▶ 1847] type exposes the following members.

Fields

	Name	Description
	Symbol [▶ 1852]	Symbol that caused the event. (Inherited from ValueChangedBaseArgs [▶ 1850].)
	UtcLocalSystemTime [▶ 1853]	Notification Time stamp of the local system (user/desktop time) in UTC (Inherited from ValueChangedBaseArgs [▶ 1850].)
	UtcRtime [▶ 1853]	Notification Time stamp of the Real time System in UTC (Inherited from ValueChangedBaseArgs [▶ 1850].)
	Value [▶ 1849]	The new received Value

Reference

ValueChangedArgs Class [▶ 1847]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.103.2.1 ValueChangedArgs.Value Field

The new received Value

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly Object Value
```

VB

```
Public ReadOnly Value As Object
```

Field Value

Type: Object

Reference

ValueChangedArgs Class [▶ 1847]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.104 ValueChangedBaseArgs Class

Event args for the RawValueChanged [▶ 1683] event.

Inheritance Hierarchy

System.Object
  System.EventArgs
    TwinCAT.TypeSystem.ValueChangedBaseArgs
      TwinCAT.TypeSystem.RawValueChangedArgs [▶ 1725]
      TwinCAT.TypeSystem.ValueChangedArgs [▶ 1847]

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ValueChangedBaseArgs : EventArgs
```

VB

```
Public Class ValueChangedBaseArgs
    Inherits EventArgs
```

The ValueChangedBaseArgs type exposes the following members.

Constructors

	Name	Description
	ValueChangedBaseArgs [▶ 1851]	Initializes a new instance of the RawValueChangedArgs [▶ 1725] class.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)

	Name	Description
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Fields

	Name	Description
	Symbol [▶ 1852]	Symbol that caused the event.
	UtcLocalSystemTime [▶ 1853]	Notification Time stamp of the local system (user/desktop time) in UTC
	UtcRtime [▶ 1853]	Notification Time stamp of the Real time System in UTC

Reference

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.104.1 ValueChangedBaseArgs Constructor

Initializes a new instance of the RawValueChangedArgs [▶ 1725] class.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected ValueChangedBaseArgs(
    ISymbol symbol,
    DateTime rtUtcTimeStamp,
    DateTime localUtcTimeStamp
)
```

VB

```
Protected Sub New (
    symbol As ISymbol,
    rtUtcTimeStamp As DateTime,
    localUtcTimeStamp As DateTime
)
```

Parameters

symbol	Type: TwinCAT.TypeSystem.ISymbol [▶ 1634] The symbol.
rtUtcTimeStamp	Type: System.DateTime The TwinCAT Real time time stamp (UTC)
localUtcTimeStamp	Type: System.DateTime The local user mode time stamp (UTC)

Reference

ValueChangedBaseArgs Class [▶ 1850]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.104.2 ValueChangedBaseArgs Methods

The ValueChangedBaseArgs [▶ 1850] type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

ValueChangedBaseArgs Class [▶ 1850]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.104.3 ValueChangedBaseArgs Fields

The ValueChangedBaseArgs [▶ 1850] type exposes the following members.

Fields

	Name	Description
	Symbol [▶ 1852]	Symbol that caused the event.
	UtcLocalSystemTime [▶ 1853]	Notification Time stamp of the local system (user/desktop time) in UTC
	UtcRtime [▶ 1853]	Notification Time stamp of the Real time System in UTC

Reference

ValueChangedBaseArgs Class [▶ 1850]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.104.3.1 ValueChangedBaseArgs.Symbol Field

Symbol that caused the event.

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly ISymbol Symbol
```

VB

```
Public ReadOnly Symbol As ISymbol
```

Field Value

Type: ISymbol [▶ 1634]

Reference

ValueChangedBaseArgs Class [▶ 1850]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.104.3.2 ValueChangedBaseArgs.UtcLocalSystemTime Field

Notification Time stamp of the local system (user/desktop time) in UTC

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly DateTime UtcLocalSystemTime
```

VB

```
Public ReadOnly UtcLocalSystemTime As DateTime
```

Field Value

Type: DateTime

Reference

ValueChangedBaseArgs Class [▶ 1850]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.8.104.3.3 ValueChangedBaseArgs.UtcRtime Field

Notification Time stamp of the Real time System in UTC

Namespace: TwinCAT.TypeSystem [▶ 1184]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public readonly DateTime UtcRtime
```

VB

```
Public ReadOnly UtcRtime As DateTime
```

Field Value

Type: DateTime

Reference

ValueChangedBaseArgs Class [▶ 1850]

TwinCAT.TypeSystem Namespace [▶ 1184]

5.9 TwinCAT.TypeSystem.Generic Namespace

Namespace for the dynamic part of the common type system.

Classes

	Class	Description
	DataTypeCollection.T. [▶ 1855]	Data type collection
	InstanceCollection.T. [▶ 1887]	Base class for IInstance [▶ 1556] object collections (abstract).
	NamespaceCollection.N, T. [▶ 1917]	Generic class for Namespace collections
	ReadOnlyDataTypeCollection.T. [▶ 1935]	ReadOnly DataType collection
	ReadOnlyInstanceCollection.T. [▶ 1942]	ReadOnly Instance collection
	ReadOnlyNamespaceCollection.N, T. [▶ 1953]	Read Only namespace collection
	ReadOnlySymbolCollection.T. [▶ 1960]	Read only symbol collection.
	SymbolCollection.T. [▶ 1964]	Interface represents a collection of ISymbol [▶ 1634] objects.
	SymbolIterator.T. [▶ 1971]	Symbol iterator object

Interfaces

	Interface	Description
	IDataTypeContainer.T. [▶ 1874]	Data Type container interface

	Interface	Description
	IInstanceCollection.T. [▶ 1877]	Generic InstanceColleciton interface.
	INamespace.T. [▶ 1885]	Namespace interface
	ISymbolCollection.T. [▶ 1911]	Interface ISymbolCollection
	ISymbolProvider.N, T, S. [▶ 1914]	Symbol provider interface

Enumerations

	Enumeration	Description
	InstanceCollectionMode [▶ 1910]	Enum InstanceCollectionMode
	SymbolIterationMask [▶ 1970]	Mask Flagset to specify filters for SymbolIterator.T. [▶ 1971].

5.9.1 DataTypeCollection.T. Class

Data type collection

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.Generic.DataTypeCollection.T.
    TwinCAT.TypeSystem.DataTypeCollection [▶ 1190]
Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class DataTypeCollection<T> : IList<T>,
    ICollection<T>, IEnumerable<T>, IEnumerable, IDataTypeContainer<T>
where T : IDataType
```

VB

```
Public Class DataTypeCollection(Of T As IDataType)
    Implements IList(Of T), ICollection(Of T),
    IEnumerable(Of T), IEnumerable, IDataTypeContainer(Of T)
```

Type Parameters

T

The DataTypeCollection.T. type exposes the following members.

Constructors

	Name	Description
	DataTypeCollection.T.. [▶ 1857]	Initializes a new instance of the DataTypeCollection [▶ 1190] class.
	DataTypeCollection.T.(IEnumerable.T.) [▶ 1858]	Initializes a new instance of the DataTypeCollection.T. class.

Properties

	Name	Description
	Count [▶ 1859]	Gets the count of contained IDataType [▶ 1517]s.
	IsReadOnly [▶ 1859]	Gets a value indicating whether this instance is read only.
	Item.Int32. [▶ 1860]	Gets or sets the IDataType [▶ 1517] at the specified index.
	Item.String. [▶ 1861]	Gets the IDataType [▶ 1517] with the specified name.

Methods

	Name	Description
	Add [▶ 1863]	Adds the specified item to the collection.
	AddRange [▶ 1863]	Adds a range of types
	AsReadOnly [▶ 1864]	Converts the DataTypeCollection.T. into a ReadOnlyCollection.T.
	Clear [▶ 1864]	Clears the collection.
	Clone [▶ 1865]	Clones this instance.
	Contains [▶ 1865]	Determines whether this DataTypeCollection [▶ 1190] contains the specified IDataType [▶ 1517].
	ContainsType [▶ 1866]	Determines whether the container contains the specified IDataType [▶ 1517].
	CopyTo [▶ 1867]	Copies the data types to the specified array, starting at the array index.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1867]	Gets the enumerator.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1868]	Determines the Index of the specified IDataType [▶ 1517].
	Insert [▶ 1868]	Inserts an IDataType [▶ 1517] into the DataTypeCollection [▶ 1190].
	LookupType [▶ 1869]	Determines the specified IDataType [▶ 1517]

	Name	Description
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1870]	Removes the specified IDataType [▶ 1517].
	RemoveAt [▶ 1870]	Removes the IDataType [▶ 1517] object at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetType [▶ 1871]	Tries to get the specified IDataType [▶ 1517] from the IDataTypeContainer.T. [▶ 1874].

Fields

	Name	Description
	list [▶ 1872]	Internal list of data types
	nameDict [▶ 1873]	Dictionary (Type Name --> DataType)
	readOnly [▶ 1873]	Indicates that the DataTypeCollection.T. is readonly

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.1 DataTypeCollection.T. Constructor

Overload List

	Name	Description
	DataTypeCollection.T.. [▶ 1857]	Initializes a new instance of the DataTypeCollection [▶ 1190] class.
	DataTypeCollection.T.(IEnumerable.T.) [▶ 1858]	Initializes a new instance of the DataTypeCollection.T. [▶ 1855] class.

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.1.1 DataTypeCollection.T. Constructor

Initializes a new instance of the DataTypeCollection [▶ 1190] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeCollection()
```

VB

```
Public Sub New
```

Reference

DataTypeCollection.T. Class [▶ 1855]

DataTypeCollection.T. Overload [▶ 1857]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.1.2 DataTypeCollection.T. Constructor (IEnumerable.T.)

Initializes a new instance of the DataTypeCollection.T. [▶ 1855] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeCollection(
    IEnumerable<T> types
)
```

VB

```
Public Sub New (
    types As IEnumerable(Of T)
)
```

Parameters

types	Type: System.Collections.Generic.IEnumerable.T [▶ 1855]. The types.

Reference

DataTypeCollection.T. Class [▶ 1855]

DataTypeCollection.T. Overload [▶ 1857]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.2 DataTypeCollection.T. Properties

The DataTypeCollection.T. [▶ 1855] generic type exposes the following members.

Properties

	Name	Description
	Count [▶ 1859]	Gets the count of contained IDataType [▶ 1517]s.
	IsReadOnly [▶ 1859]	Gets a value indicating whether this instance is read only.
	Item.Int32. [▶ 1860]	Gets or sets the IDataType [▶ 1517] at the specified index.
	Item.String. [▶ 1861]	Gets the IDataType [▶ 1517] with the specified name.

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.2.1 DataTypeCollection.T..Count Property

Gets the count of contained IDataType [▶ 1517]s.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Count { get; }
```

VB

```
Public ReadOnly Property Count As Integer
    Get
```

Property Value

Type: Int32
The count.

Implements

ICollection.T..Count

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.2.2 DataTypeCollection.T..IsReadOnly Property

Gets a value indicating whether this instance is read only.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is read only; otherwise, false.

Implements

ICollection.T..IsReadOnly

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.2.3 DataTypeCollection.T..Item Property

Overload List

	Name	Description
	Item.Int32. [▶ 1860]	Gets or sets the IDataType [▶ 1517] at the specified index.
	Item.String. [▶ 1861]	Gets the IDataType [▶ 1517] with the specified name.

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

DataTypeCollection.T..Item Property (Int32)

Gets or sets the IDataType [▶ 1517] at the specified index.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T this[
    int index
] { get; set; }
```

VB

```
Public Default Property Item (
    index As Integer
) As T
    Get
    Set
```

Parameters

index — Type: System.Int32
The index.

Return Value

Type: T [▶ 1855]
T.

Implements

IList.T..Item.Int32.

Exceptions

Exception	Condition
NotImplementedException	

Reference

DataTypeCollection.T. Class [▶ 1855]

Item Overload [▶ 1860]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

DataTypeCollection.T..Item Property (String)

Gets the IDataType [▶ 1517] with the specified name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T this[
    string name
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    name As String
) As T
    Get
```

Parameters

name	Type: System.String The name.

Return Value

Type: T [▶ 1855]
T.

Implements

IDataTypeContainer.T..Item.String. [▶ 1875]

Reference

DataTypeCollection.T. Class [▶ 1855]

Item Overload [▶ 1860]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3 DataTypeCollection.T. Methods

The DataTypeCollection.T. [▶ 1855] generic type exposes the following members.

Methods

	Name	Description
	Add [▶ 1863]	Adds the specified item to the collection.
	AddRange [▶ 1863]	Adds a range of types
	AsReadOnly [▶ 1864]	Converts the DataTypeCollection.T. [▶ 1855] into a ReadOnlyCollection.T.
	Clear [▶ 1864]	Clears the collection.
	Clone [▶ 1865]	Clones this instance.
	Contains [▶ 1865]	Determines whether this DataTypeCollection [▶ 1190] contains the specified IDataType [▶ 1517].
	ContainsType [▶ 1866]	Determines whether the container contains the specified IDataType [▶ 1517].
	CopyTo [▶ 1867]	Copies the data types to the specified array, starting at the array index.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1867]	Gets the enumerator.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1868]	Determines the Index of the specified IDataType [▶ 1517].
	Insert [▶ 1868]	Inserts an IDataType [▶ 1517] into the DataTypeCollection [▶ 1190].
	LookupType [▶ 1869]	Determines the specified IDataType [▶ 1517]
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1870]	Removes the specified IDataType [▶ 1517].
	RemoveAt [▶ 1870]	Removes the IDataType [▶ 1517] object at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetType [▶ 1871]	Tries to get the specified IDataType [▶ 1517] from the IDataTypeContainer.T. [▶ 1874].

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.1 DataTypeCollection.T..Add Method

Adds the specified item to the collection.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Add(
    T item
)
```

VB

```
Public Sub Add (
    item As T
)
```

Parameters

item	Type: T [▶ 1855] The item.

Implements

ICollection.T..Add(T)

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.2 DataTypeCollection.T..AddRange Method

Adds a range of types

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void AddRange(
    IEnumerable<T> types
)
```

VB

```
Public Sub AddRange (
    types As IEnumerable(Of T)
)
```

Parameters

types	Type: System.Collections.Generic.IEnumerable.T [▶ 1855]. The types.

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.3 DataTypeCollection.T..AsReadOnly Method

Converts the DataTypeCollection.T. [▶ 1855] into a ReadOnlyCollection.T.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyDataTypeCollection<T> AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlyDataTypeCollection(Of T)
```

Return Value

Type: ReadOnlyDataTypeCollection [▶ 1935].T [▶ 1855].
ReadOnlyDataTypeCollection<T>.

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.4 DataTypeCollection.T..Clear Method

Clears the collection.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Clear()
```

VB

```
Public Sub Clear
```

Implements

ICollection.T..Clear.

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.5 DataTypeCollection.T..Clone Method

Clones this instance.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public DataTypeCollection<T> Clone()
```

VB

```
Public Function Clone As DataTypeCollection(Of T)
```

Return Value

Type: DataTypeCollection [▶ 1855].T [▶ 1855].
DataTypeCollection<T>.

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.6 DataTypeCollection.T..Contains Method

Determines whether this DataTypeCollection [▶ 1190] contains the specified IDataType [▶ 1517].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    T item
)
```

VB

```
Public Function Contains (
    item As T
) As Boolean
```

Parameters

item	Type: T [▶ 1855] The item.

Return Value

Type: Boolean
true if [contains] [the specified item]; otherwise, false.

Implements

ICollection.T..Contains(T)

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.7 DataTypeCollection.T..ContainsType Method

Determines whether the container contains the specified IDataType [▶ 1517].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool ContainsType(
    string name
)
```

VB

```
Public Function ContainsType (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name.

Return Value

Type: Boolean
true if contained; otherwise, false.

Implements

IDataTypeContainer.T..ContainsType(String) [▶ 1875]

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.8 DataTypeCollection.T..CopyTo Method

Copies the data types to the specified array, starting at the array index.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void CopyTo(
    T[] array,
    int arrayIndex
)
```

VB

```
Public Sub CopyTo (
    array As T(),
    arrayIndex As Integer
)
```

Parameters

array	Type: .T [▶ 1855]. The array.
arrayIndex	Type: System.Int32 Index of the array.

Implements

ICollection.T..CopyTo(.T., Int32)

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.9 DataTypeCollection.T..GetEnumerator Method

Gets the enumerator.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumerator<T> GetEnumerator()
```

VB

```
Public Function GetEnumerator As IEnumerator(Of T)
```

Return Value

Type: IEnumerator.T [▶ 1855].
A IEnumerator.T. that can be used to iterate through the collection.

Implements

IEnumerable.T..GetEnumerator.

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.10 DataTypeCollection.T..IndexOf Method

Determines the Index of the specified IDataType [▶ 1517].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int IndexOf(
    T item
)
```

VB

```
Public Function IndexOf (
    item As T
) As Integer
```

Parameters

item	Type: T [▶ 1855] The item.

Return Value

Type: Int32
The index of item if found in the list; otherwise, -1.

Implements

IList.T..IndexOf(T)

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.11 DataTypeCollection.T..Insert Method

Inserts an IDataType [▶ 1517] into the DataTypeCollection [▶ 1190].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Insert(
    int index,
    T item
)
```

VB

```
Public Sub Insert (
    index As Integer,
    item As T
)
```

Parameters

index	Type: System.Int32 The index.
item	Type: T [▶ 1855] The item.

Implements

IList.T..Insert(Int32, T)

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.12 DataTypeCollection.T..LookupType Method

Determines the specified IDataType [▶ 1517]

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T LookupType(
    string name
)
```

VB

```
Public Function LookupType (
    name As String
) As T
```

Parameters

name	Type: System.String The name.

Return Value

Type: T [▶ 1855]
The IDataType [▶ 1517] if found, otherwise NULL

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.13 DataTypeCollection.T..Remove Method

Removes the specified IDataType [▶ 1517].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Remove(
    T item
)
```

VB

```
Public Function Remove (
    item As T
) As Boolean
```

Parameters

item	Type: T [▶ 1855] The item.

Return Value

Type: Boolean
true if item was successfully removed from the ICollection.T.; otherwise, false. This method also returns false if item is not found in the original ICollection.T..

Implements

ICollection.T..Remove(T)

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.14 DataTypeCollection.T..RemoveAt Method

Removes the IDataType [▶ 1517] object at the specified index.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void RemoveAt(
    int index
)
```

VB

```
Public Sub RemoveAt (
    index As Integer
)
```

Parameters

index	Type: System.Int32 The index.

Implements

IList.T..RemoveAt(Int32)

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.3.15 DataTypeCollection.T..TryGetType Method

Tries to get the specified IDataType [▶ 1517] from the IDataTypeContainer.T. [▶ 1874].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetType(
    string name,
    out T type
)
```

VB

```
Public Function TryGetType (
    name As String,
    <OutAttribute> ByRef type As T
) As Boolean
```

Parameters

name	Type: System.String The name.
type	Type: T [▶ 1855]. The type (Out parameter)

Return Value

Type: Boolean
true if found

Implements

IDataTypeContainer.T..TryGetType(String, T.) [▶ 1876]

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.4 DataTypeCollection.T. Fields

The DataTypeCollection.T. [▶ 1855] generic type exposes the following members.

Fields

	Name	Description
	list [▶ 1872]	Internal list of data types
	nameDict [▶ 1873]	Dictionary (Type Name --> DataType)
	readOnly [▶ 1873]	Indicates that the DataTypeCollection.T. [▶ 1855] is readonly

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.4.1 DataTypeCollection.T..list Field

Internal list of data types

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected List<T> list
```

VB

```
Protected list As List(Of T)
```

Field Value

Type: List.T [▶ 1855].

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.4.2 DataTypeCollection.T..nameDict Field

Dictionary (Type Name --> DataType)

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected Dictionary<string, T> nameDict
```

VB

```
Protected nameDict As Dictionary(Of String, T)
```

Field Value

Type: Dictionary.String, T [▶ 1855].

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.1.4.3 DataTypeCollection.T..readOnly Field

Indicates that the DataTypeCollection.T. [▶ 1855] is readonly

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected bool readOnly
```

VB

```
Protected readOnly As Boolean
```

Field Value

Type: Boolean

Reference

DataTypeCollection.T. Class [▶ 1855]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.2 IDataTypeContainer.T. Interface

Data Type container interface

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IDataTypeContainer<T>
where T : IDataType
```

VB

```
Public Interface IDataTypeContainer(Of T As IDataType)
```

Type Parameters

T	Data Type type.

The IDataTypeContainer.T. type exposes the following members.

Properties

	Name	Description
	Item [▶ 1875]	Gets the IDataType [▶ 1517] with the specified name.

Methods

	Name	Description
	ContainsType [▶ 1875]	Determines whether the container contains the specified IDataType [▶ 1517].
	TryGetType [▶ 1876]	Tries to get the specified IDataType [▶ 1517] from the IDataTypeContainer.T..

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.2.1 IDataTypeContainer.T. Properties

The IDataTypeContainer.T. [▶ 1874] generic type exposes the following members.

Properties

	Name	Description
	Item [▶ 1875]	Gets the IDataType [▶ 1517] with the specified name.

Reference

IDataTypeContainer.T. Interface [▶ 1874]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.2.1.1 IDataTypeContainer.T..Item Property

Gets the IDataType [▶ 1517] with the specified name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
T this[
    string name
] { get; }
```

VB

```
ReadOnly Default Property Item (
    name As String
) As T
    Get
```

Parameters

name	Type: System.String The name.

Return Value

Type: T [▶ 1874]
T.

Reference

IDataTypeContainer.T. Interface [▶ 1874]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.2.2 IDataTypeContainer.T. Methods

The IDataTypeContainer.T. [▶ 1874] generic type exposes the following members.

Methods

	Name	Description
	ContainsType [▶ 1875]	Determines whether the container contains the specified IDataType [▶ 1517].
	TryGetType [▶ 1876]	Tries to get the specified IDataType [▶ 1517] from the IDataTypeContainer.T. [▶ 1874].

Reference

IDataTypeContainer.T. Interface [▶ 1874]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.2.2.1 IDataTypeContainer.T..ContainsType Method

Determines whether the container contains the specified IDataType [▶ 1517].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool ContainsType(
    string name
)
```

VB

```
Function ContainsType (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name.

Return Value

Type: Boolean
true if contained; otherwise, false.

Reference

IDataTypeContainer.T. Interface [▶ 1874]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.2.2.2 IDataTypeContainer.T..TryGetType Method

Tries to get the specified IDataType [▶ 1517] from the IDataTypeContainer.T. [▶ 1874].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TryGetType(
    string name,
    out T type
)
```

VB

```
Function TryGetType (
    name As String,
    <OutAttribute> ByRef type As T
) As Boolean
```

Parameters

name	Type: System.String The name.
type	Type: T [▶ 1874]. The type (Out parameter)

Return Value

Type: Boolean
true if found

Reference

IDataTypeContainer.T. Interface [▶ 1874]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.3 IInstanceCollection.T. Interface

Generic InstanceColleciton interface.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface IInstanceCollection<T> : IList<T>,
    ICollection<T>, IEnumerable<T>, IEnumerable
where T : IInstance
```

VB

```
Public Interface IInstanceCollection(Of T As IInstance)
    Inherits IList(Of T), ICollection(Of T), IEnumerable(Of T),
    IEnumerable
```

Type Parameters

T

The IInstanceCollection.T. type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ICollection.T.. (Inherited from ICollection.T..)
	IsReadOnly	Gets a value indicating whether the ICollection.T. is read-only. (Inherited from ICollection.T..)
	Item.Int32.	Gets or sets the element at the specified index. (Inherited from IList.T..)
	Item.String. [▶ 1879]	Gets the IInstance [▶ 1556] with the specified instance path.
	Mode [▶ 1880]	Gets the InstanceCollectionMode [▶ 1910].

Methods

	Name	Description
	Add	Adds an item to the ICollection.T.. (Inherited from ICollection.T..)
	Clear	Removes all items from the ICollection.T.. (Inherited from ICollection.T..)

	Name	Description
	Contains(T)	Determines whether the ICollection.T. contains a specific value. (Inherited from ICollection.T..)
	Contains(String) [▶ 1881]	Determines whether this collection contains an instance with the specified instance path.
	ContainsName [▶ 1882]	Determines whether this collection contains an instance with the specified instance name.
	CopyTo	Copies the elements of the ICollection.T. to an Array, starting at a particular Array index. (Inherited from ICollection.T..)
	GetEnumerator	Returns an enumerator that iterates through the collection. (Inherited from IEnumerable.T..)
	GetInstance [▶ 1883]	Gets the IInstance [▶ 1556]by instance path.
	GetInstanceByName [▶ 1883]	Gets the IInstance [▶ 1556] by instance name.
	IndexOf	Determines the index of a specific item in the IList.T.. (Inherited from IList.T..)
	Insert	Inserts an item to the IList.T. at the specified index. (Inherited from IList.T..)
	Remove	Removes the first occurrence of a specific object from the ICollection.T.. (Inherited from ICollection.T..)
	RemoveAt	Removes the IList.T. item at the specified index. (Inherited from IList.T..)
	TryGetInstance [▶ 1884]	Tries to get the specified instance.
	TryGetInstanceByName [▶ 1884]	Tries to get the specified instance by name.

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.3.1 IInstanceCollection.T. Properties

The IInstanceCollection.T. [▶ 1877] generic type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ICollection.T.. (Inherited from ICollection.T [▶ 1877]..)
	IsReadOnly	Gets a value indicating whether the ICollection.T. is read-only. (Inherited from ICollection.T [▶ 1877]..)
	Item.Int32.	Gets or sets the element at the specified index. (Inherited from IList.T [▶ 1877]..)
	Item.String. [▶ 1879]	Gets the IInstance [▶ 1556] with the specified instance path.
	Mode [▶ 1880]	Gets the InstanceCollectionMode [▶ 1910].

Reference

IInstanceCollection.T. Interface [▶ 1877]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.3.1.1 IInstanceCollection.T..Item Property

Overload List

	Name	Description
	Item.Int32.	Gets or sets the element at the specified index. (Inherited from IList.T [▶ 1877]..)
	Item.String. [▶ 1879]	Gets the IInstance [▶ 1556] with the specified instance path.

Reference

IInstanceCollection.T. Interface [▶ 1877]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

IInstanceCollection.T..Item Property (String)

Gets the IInstance [▶ 1556] with the specified instance path.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
T this[
    string instancePath
] { get; }
```

VB

```
ReadOnly Default Property Item (
    instancePath As String
) As T
    Get
```

Parameters

instancePath	Type: System.String

Property Value

Type: T [▶ 1877]

Reference

IInstanceCollection.T. Interface [▶ 1877]

Item Overload [▶ 1879]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.3.1.2 IInstanceCollection.T..Mode Property

Gets the InstanceCollectionMode [▶ 1910].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
InstanceCollectionMode Mode { get; }
```

VB

```
ReadOnly Property Mode As InstanceCollectionMode
    Get
```

Property Value

Type: InstanceCollectionMode [▶ 1910]
The mode.

Reference

IInstanceCollection.T. Interface [▶ 1877]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.3.2 IInstanceCollection.T. Methods

The IInstanceCollection.T. [▶ 1877] generic type exposes the following members.

Methods

	Name	Description
	Add	Adds an item to the ICollection.T.. (Inherited from ICollection.T [▶ 1877]..)
	Clear	Removes all items from the ICollection.T.. (Inherited from ICollection.T [▶ 1877]..)
	Contains(T)	Determines whether the ICollection.T. contains a specific value. (Inherited from ICollection.T [▶ 1877]..)
	Contains(String) [▶ 1881]	Determines whether this collection contains an instance with the specified instance path.
	ContainsName [▶ 1882]	Determines whether this collection contains an instance with the specified instance name.
	CopyTo	Copies the elements of the ICollection.T. to an Array, starting at a particular Array index. (Inherited from ICollection.T [▶ 1877]..)
	GetEnumerator	Returns an enumerator that iterates through the collection. (Inherited from IEnumerable.T [▶ 1877]..)
	GetInstance [▶ 1883]	Gets the IInstance [▶ 1556]by instance path.
	GetInstanceByName [▶ 1883]	Gets the IInstance [▶ 1556] by instance name.

	Name	Description
	IndexOf	Determines the index of a specific item in the IList.T.. (Inherited from IList.T [▶ 1877]..)
	Insert	Inserts an item to the IList.T. at the specified index. (Inherited from IList.T [▶ 1877]..)
	Remove	Removes the first occurrence of a specific object from the ICollection.T.. (Inherited from ICollection.T [▶ 1877]..)
	RemoveAt	Removes the IList.T. item at the specified index. (Inherited from IList.T [▶ 1877]..)
	TryGetInstance [▶ 1884]	Tries to get the specified instance.
	TryGetInstanceByName [▶ 1884]	Tries to get the specified instance by name.

Reference

IInstanceCollection.T. Interface [▶ 1877]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.3.2.1 IInstanceCollection.T..Contains Method

Overload List

	Name	Description
	Contains(T)	Determines whether the ICollection.T. contains a specific value. (Inherited from ICollection.T [▶ 1877]..)
	Contains(String) [▶ 1881]	Determines whether this collection contains an instance with the specified instance path.

Reference

IInstanceCollection.T. Interface [▶ 1877]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

IInstanceCollection.T..Contains Method (String)

Determines whether this collection contains an instance with the specified instance path.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool Contains(
    string instancePath
)
```

VB

```
Function Contains (
    instancePath As String
) As Boolean
```

Parameters

instancePath	Type: System.String The instance path.

Return Value

Type: Boolean
true if this collection contains the specified instance path; otherwise, false.

Reference

IInstanceCollection.T. Interface [▶ 1877]

Contains Overload [▶ 1881]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.3.2.2 IInstanceCollection.T..ContainsName Method

Determines whether this collection contains an instance with the specified instance name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool ContainsName(
    string instanceName
)
```

VB

```
Function ContainsName (
    instanceName As String
) As Boolean
```

Parameters

instanceName	Type: System.String Name of the instance.

Return Value

Type: Boolean
true if this collection contains the specified instance path; otherwise, false.

Reference

IInstanceCollection.T. Interface [▶ 1877]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.3.2.3 IInstanceCollection.T..GetInstance Method

Gets the IInstance [▶ 1556]by instance path.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
T GetInstance(
    string instancePath
)
```

VB

```
Function GetInstance (
    instancePath As String
) As T
```

Parameters

instancePath	Type: System.String The instance path.

Return Value

Type: T [▶ 1877]
T.

Reference

IInstanceCollection.T. Interface [▶ 1877]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.3.2.4 IInstanceCollection.T..GetInstanceByName Method

Gets the IInstance [▶ 1556] by instance name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
IList<T> GetInstanceByName(
    string instanceName
)
```

VB

```
Function GetInstanceByName (
    instanceName As String
) As IList(Of T)
```

Parameters

instanceName	Type: System.String Name of the instance.

Return Value

Type: IList.T [▶ 1877].
IList<T>.

Reference

IInstanceCollection.T. Interface [▶ 1877]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.3.2.5 IInstanceCollection.T..TryGetInstance Method

Tries to get the specified instance.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TryGetInstance(
    string instancePath,
    out T symbol
)
```

VB

```
Function TryGetInstance (
    instancePath As String,
    <OutAttribute> ByRef symbol As T
) As Boolean
```

Parameters

instancePath	Type: System.String The instance path.
symbol	Type: T [▶ 1877]. The symbol.

Return Value

Type: Boolean
true if the IInstance [▶ 1556] is found; otherwise, false

Reference

IInstanceCollection.T. Interface [▶ 1877]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.3.2.6 IInstanceCollection.T..TryGetInstanceByName Method

Tries to get the specified instance by name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
bool TryGetInstanceByName(
    string instanceName,
    out IList<T> symbols
)
```

VB

```
Function TryGetInstanceByName (
    instanceName As String,
    <OutAttribute> ByRef symbols As IList(Of T)
) As Boolean
```

Parameters

instanceName	Type: System.String Name of the instance.
symbols	Type: System.Collections.Generic.IList.T [▶ 1877].. The found symbols.

Return Value

Type: Boolean
true if the IInstance [▶ 1556] is found; otherwise, false

Reference

IInstanceCollection.T. Interface [▶ 1877]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.4 INamespace.T. Interface

Namespace interface

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface INamespace<T>
where T : IDataType
```

VB

```
Public Interface INamespace(Of T As IDataType)
```

Type Parameters

T	DataType class used within this Namespace interface

The INamespace.T. type exposes the following members.

Properties

	Name	Description
	DataTypes [▶ 1886]	Data types organized by the INamespace.T.
	Name [▶ 1887]	Gets the name/ identifier of the Namespace

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.4.1 INamespace.T. Properties

The INamespace.T. [▶ 1885] generic type exposes the following members.

Properties

	Name	Description
	DataTypes [▶ 1886]	Data types organized by the INamespace.T. [▶ 1885]
	Name [▶ 1887]	Gets the name/ identifier of the Namespace

Reference

INamespace.T. Interface [▶ 1885]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.4.1.1 INamespace.T..DataTypes Property

Data types organized by the INamespace.T. [▶ 1885]

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyDataTypeCollection<T> DataTypes { get; }
```

VB

```
ReadOnly Property DataTypes As ReadOnlyDataTypeCollection(Of T)
    Get
```

Property Value

Type: ReadOnlyDataTypeCollection [▶ 1935].T [▶ 1885].
The data types.

Reference

INamespace.T. Interface [▶ 1885]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.4.1.2 INamespace.T..Name Property

Gets the name/ identifier of the Namespace

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string Name { get; }
```

VB

```
ReadOnly Property Name As String
    Get
```

Property Value

Type: String
The name.

Reference

INamespace.T. Interface [▶ 1885]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5 InstanceCollection.T. Class

Base class for IInstance [▶ 1556] object collections (abstract).

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.Generic.InstanceCollection.T.
    TwinCAT.TypeSystem.FieldCollection [▶ 1484]
    TwinCAT.TypeSystem.Generic.SymbolCollection.T. [▶ 1964]
    TwinCAT.TypeSystem.MemberCollection [▶ 1717]
Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public abstract class InstanceCollection<T> : IList<T>,
    ICollection<T>, IEnumerable<T>, IEnumerable, IInstanceCollection<T>
where T : class, IInstance
```

VB

```
Public MustInherit Class InstanceCollection(Of T As {Class, IInstance})
    Implements IList(Of T), ICollection(Of T),
    IEnumerable(Of T), IEnumerable, IInstanceCollection(Of T)
```

Type Parameters

T

The InstanceCollection.T. type exposes the following members.

Constructors

	Name	Description
	InstanceCollection.T .(InstanceCollectionMode) [▶ 1890]	Initializes a new instance of the InstanceCollection.T. class.
	InstanceCollection.T .(IEnumerable.T., InstanceCollectionMode) [▶ 1890]	Initializes a new instance of the InstanceCollection.T. class.

Properties

	Name	Description
	Count [▶ 1891]	Gets the collection count.
	IsReadOnly [▶ 1892]	Gets a value indicating whether this instance is read only.
	Item.Int32. [▶ 1893]	Gets or sets the IInstance [▶ 1556] at the specified index.
	Item.String. [▶ 1894]	Gets the IInstance [▶ 1556] with the specified instance path.
	Mode [▶ 1894]	Gets the InstanceCollectionMode [▶ 1910].

Methods

	Name	Description
	Add [▶ 1896]	Adds the specified item.
	AddRange [▶ 1897]	Adds the specified items to this collection.
	AsReadOnly [▶ 1897]	Converts the InstanceCollection.T. to an ReadOnlyInstanceCollection.T. [▶ 1942]
	Clear [▶ 1898]	Clears this instance.
	Contains(String) [▶ 1899]	Determines whether this collection contains an IInstance [▶ 1556] with the specified InstanceName / InstancePath
	Contains(T) [▶ 1899]	Determines whether this collection contains the specified IInstance [▶ 1556]
	ContainsName [▶ 1900]	Determines whether the specified instance name contains name.
	CopyTo [▶ 1901]	Copies this InstanceCollection.T. to the specified array.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1902]	Gets the enumerator.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1902]	Gets the IInstance [▶ 1556]by instance path.

	Name	Description
	GetInstanceByName [▶ 1903]	Gets the name of the instance by.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1904]	Determines the index of the specified IInstance [▶ 1556].
	Insert [▶ 1905]	Inserts the specified IInstance [▶ 1556] at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1905]	Removes the specified item.
	RemoveAt [▶ 1906]	Removes the IInstance [▶ 1556] at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1907]	Tries to get the IInstance [▶ 1556]. of the specified path.
	TryGetInstanceByName [▶ 1908]	Tries to get Instnaces by name.

Fields

	Name	Description
	_list [▶ 1909]	The _list
	_pathDict [▶ 1909]	The _path dictionary
	mode [▶ 1910]	The mode this InstanceCollection.T. is working in.

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.1 InstanceCollection.T. Constructor

Overload List

	Name	Description
	InstanceCollection.T. (InstanceCollectionMode) [▶ 1890]	Initializes a new instance of the InstanceCollection.T. [▶ 1887] class.
	InstanceCollection.T.(IEnumerable.T., InstanceCollectionMode) [▶ 1890]	Initializes a new instance of the InstanceCollection.T. [▶ 1887] class.

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.1.1 InstanceCollection.T. Constructor (InstanceCollectionMode)

Initializes a new instance of the InstanceCollection.T. [▶ 1887] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected InstanceCollection(
    InstanceCollectionMode mode
)
```

VB

```
Protected Sub New (
    mode As InstanceCollectionMode
)
```

Parameters

mode	Type: TwinCAT.TypeSystem.Generic.InstanceCollectionMode [▶ 1910] The mode.

Reference

InstanceCollection.T. Class [▶ 1887]

InstanceCollection.T. Overload [▶ 1889]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.1.2 InstanceCollection.T. Constructor (IEnumerable.T., InstanceCollectionMode)

Initializes a new instance of the InstanceCollection.T. [▶ 1887] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected InstanceCollection(
    IEnumerable<T> coll,
    InstanceCollectionMode mode
)
```

VB

```
Protected Sub New (
    coll As IEnumerable(Of T),
    mode As InstanceCollectionMode
)
```

Parameters

coll	Type: System.Collections.Generic.IEnumerable.T [▶ 1887]. The copy collection
mode	Type: TwinCAT.TypeSystem.Generic.InstanceCollectionMode [▶ 1910] The mode.

Reference

InstanceCollection.T. Class [▶ 1887]

InstanceCollection.T. Overload [▶ 1889]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.2 InstanceCollection.T. Properties

The InstanceCollection.T. [▶ 1887] generic type exposes the following members.

Properties

	Name	Description
	Count [▶ 1891]	Gets the collection count.
	IsReadOnly [▶ 1892]	Gets a value indicating whether this instance is read only.
	Item.Int32. [▶ 1893]	Gets or sets the IInstance [▶ 1556] at the specified index.
	Item.String. [▶ 1894]	Gets the IInstance [▶ 1556] with the specified instance path.
	Mode [▶ 1894]	Gets the InstanceCollectionMode [▶ 1910].

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.2.1 InstanceCollection.T..Count Property

Gets the collection count.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Count { get; }
```

VB

```
Public ReadOnly Property Count As Integer
    Get
```

Property Value

Type: Int32
The count.

Implements

ICollection.T..Count

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.2.2 InstanceCollection.T..IsReadOnly Property

Gets a value indicating whether this instance is read only.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Property Value

Type: Boolean
true if this instance is read only; otherwise, false.

Implements

ICollection.T..IsReadOnly

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.2.3 InstanceCollection.T..Item Property

Overload List

	Name	Description
	Item.Int32. [▶ 1893]	Gets or sets the IInstance [▶ 1556] at the specified index.
	Item.String. [▶ 1894]	Gets the IInstance [▶ 1556] with the specified instance path.

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

InstanceCollection.T..Item Property (Int32)

Gets or sets the IInstance [▶ 1556] at the specified index.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T this[
    int index
] { get; set; }
```

VB

```
Public Default Property Item (
    index As Integer
) As T
    Get
    Set
```

Parameters

index	Type: System.Int32 The index.

Return Value

Type: T [▶ 1887]
T.

Implements

IList.T..Item.Int32.

Exceptions

Exception	Condition
NotImplementedException	

Reference

InstanceCollection.T. Class [▶ 1887]

Item Overload [▶ 1892]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

InstanceCollection.T..Item Property (String)

Gets the IInstance [▶ 1556] with the specified instance path.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T this[
    string instanceSpecifier
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    instanceSpecifier As String
) As T
    Get
```

Parameters

instanceSpecifier	Type: System.String The instance path or Instance Name (dependent of Mode [▶ 1894] setting)

Return Value

Type: T [▶ 1887]
T.

Implements

IInstanceCollection.T..Item.String. [▶ 1879]

Exceptions

Exception	Condition
ArgumentNullException	
ArgumentException	

Remarks

Dependent what this InstanceCollection.T. [▶ 1887] contains configured by the InstanceCollectionMode [▶ 1910] the instance specifier should be the InstanceName [▶ 1559] or the InstancePath [▶ 1560].

Reference

InstanceCollection.T. Class [▶ 1887]

Item Overload [▶ 1892]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.2.4 InstanceCollection.T..Mode Property

Gets the InstanceCollectionMode [▶ 1910].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public InstanceCollectionMode Mode { get; }
```

VB

```
Public ReadOnly Property Mode As InstanceCollectionMode
    Get
```

Property Value

Type: InstanceCollectionMode [▶ 1910]
The mode.

Implements

IInstanceCollection.T..Mode [▶ 1880]

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3 InstanceCollection.T. Methods

The InstanceCollection.T. [▶ 1887] generic type exposes the following members.

Methods

	Name	Description
	Add [▶ 1896]	Adds the specified item.
	AddRange [▶ 1897]	Adds the specified items to this collection.
	AsReadOnly [▶ 1897]	Converts the InstanceCollection.T. [▶ 1887] to an ReadOnlyInstanceCollection.T. [▶ 1942]
	Clear [▶ 1898]	Clears this instance.
	Contains(String) [▶ 1899]	Determines whether this collection contains an IInstance [▶ 1556] with the specified InstanceName / InstancePath
	Contains(T) [▶ 1899]	Determines whether this collection contains the specified IInstance [▶ 1556]
	ContainsName [▶ 1900]	Determines whether the specified instance name contains name.
	CopyTo [▶ 1901]	Copies this InstanceCollection.T. [▶ 1887] to the specified array.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	GetEnumerator [▶ 1902]	Gets the enumerator.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1902]	Gets the IInstance [▶ 1556]by instance path.
	GetInstanceByName [▶ 1903]	Gets the name of the instance by.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1904]	Determines the index of the specified IInstance [▶ 1556].
	Insert [▶ 1905]	Inserts the specified IInstance [▶ 1556] at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1905]	Removes the specified item.
	RemoveAt [▶ 1906]	Removes the IInstance [▶ 1556] at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1907]	Tries to get the IInstance [▶ 1556]. of the specified path.
	TryGetInstanceByName [▶ 1908]	Tries to get Instnaces by name.

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.1 InstanceCollection.T..Add Method

Adds the specified item.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Add(
    T item
)
```

VB

```
Public Sub Add (
    item As T
)
```

Parameters

item	Type: T [▶ 1887] The item.

Implements

ICollection.T..Add(T)

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.2 InstanceCollection.T..AddRange Method

Adds the specified items to this collection.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void AddRange(
    IEnumerable<T> items
)
```

VB

```
Public Sub AddRange (
    items As IEnumerable(Of T)
)
```

Parameters

items	Type: System.Collections.Generic.IEnumerable.T [▶ 1887]. The items.

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.3 InstanceCollection.T..AsReadOnly Method

Converts the InstanceCollection.T. [▶ 1887] to an ReadOnlyInstanceCollection.T. [▶ 1942]

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyInstanceCollection<T> AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlyInstanceCollection(Of T)
```

Return Value

Type: ReadOnlyInstanceCollection [▶ 1942].T [▶ 1887].
ReadOnlyInstanceCollection<T>.

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.4 InstanceCollection.T..Clear Method

Clears this instance.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Clear()
```

VB

```
Public Sub Clear
```

Implements

ICollection.T..Clear.

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.5 InstanceCollection.T..Contains Method

Overload List

	Name	Description
	Contains(String) [▶ 1899]	Determines whether this collection contains an IInstance [▶ 1556] with the specified InstanceName / InstancePath
	Contains(T) [▶ 1899]	Determines whether this collection contains the specified IInstance [▶ 1556]

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

InstanceCollection.T..Contains Method (String)

Determines whether this collection contains an IInstance [▶ 1556] with the specified InstanceName / InstancePath

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    string instanceSpecifier
)
```

VB

```
Public Function Contains (
    instanceSpecifier As String
) As Boolean
```

Parameters

instanceSpecifier	Type: System.String The instance path or Instance Name (dependent of Mode [▶ 1894] setting)

Return Value

Type: Boolean
true if [contains] [the specified instance path]; otherwise, false.

Implements

IInstanceCollection.T..Contains(String) [▶ 1881]

Exceptions

Exception	Condition
ArgumentNullException	instancePath
ArgumentException	

Reference

InstanceCollection.T. Class [▶ 1887]

Contains Overload [▶ 1898]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

InstanceCollection.T..Contains Method (T)

Determines whether this collection contains the specified IInstance [▶ 1556]

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    T item
)
```

VB

```
Public Function Contains (
    item As T
) As Boolean
```

Parameters

item	Type: T [▶ 1887] The item.

Return Value

Type: Boolean
true if [contains] [the specified item]; otherwise, false.

Implements

ICollection.T..Contains(T)

Reference

InstanceCollection.T. Class [▶ 1887]

Contains Overload [▶ 1898]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.6 InstanceCollection.T..ContainsName Method

Determines whether the specified instance name contains name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool ContainsName(
    string instanceName
)
```

VB

```
Public Function ContainsName (
    instanceName As String
) As Boolean
```

Parameters

instanceName	Type: System.String Name of the instance.

Return Value

Type: Boolean
true if the specified instance name contains name; otherwise, false.

Implements

IInstanceCollection.T..ContainsName(String) [▶ 1882]

Exceptions

Exception	Condition
NotImplementedException	

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.7 InstanceCollection.T..CopyTo Method

Copies this InstanceCollection.T. [▶ 1887] to the specified array.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void CopyTo(
    T[] array,
    int arrayIndex
)
```

VB

```
Public Sub CopyTo (
    array As T(),
    arrayIndex As Integer
)
```

Parameters

array	Type: .T [▶ 1887]. The array.
arrayIndex	Type: System.Int32 Index of the array.

Implements

ICollection.T..CopyTo(.T., Int32)

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.8 InstanceCollection.T..GetEnumerator Method

Gets the enumerator.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumerator<T> GetEnumerator()
```

VB

```
Public Function GetEnumerator As IEnumerator(Of T)
```

Return Value

Type: IEnumerator.T [▶ 1887].
A IEnumerator.T. that can be used to iterate through the collection.

Implements

IEnumerable.T..GetEnumerator.

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.9 InstanceCollection.T..GetInstance Method

Gets the IInstance [▶ 1556]by instance path.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T GetInstance(
    string instanceSpecifier
)
```

VB

```
Public Function GetInstance (
    instanceSpecifier As String
) As T
```

Parameters

instanceSpecifier	Type: System.String The instance path or Instance Name (dependent of Mode [▶ 1894] setting)

Return Value

Type: T [▶ 1887]
T.

Implements

IInstanceCollection.T..GetInstance(String) [▶ 1883]

Exceptions

Exception	Condition
ArgumentException	Path not found!;instancePath

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.10 InstanceCollection.T..GetInstanceByName Method

Gets the name of the instance by.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IList<T> GetInstanceByName(
    string instanceName
)
```

VB

```
Public Function GetInstanceByName (
    instanceName As String
) As IList(Of T)
```

Parameters

instanceName	Type: System.String Name of the instance.

Return Value

Type: IList.T [▶ 1887].
IList<T>.

Implements

IInstanceCollection.T..GetInstanceByName(String) [▶ 1883]

Exceptions

Exception	Condition
ArgumentException	Name not found!;instanceName

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.11 InstanceCollection.T..IndexOf Method

Determines the index of the specified IInstance [▶ 1556].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int IndexOf(
    T item
)
```

VB

```
Public Function IndexOf (
    item As T
) As Integer
```

Parameters

item	Type: T [▶ 1887] The item.

Return Value

Type: Int32
The index of item if found in the list; otherwise, -1.

Implements

IList.T..IndexOf(T)

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.12 InstanceCollection.T..Insert Method

Inserts the specified IInstance [▶ 1556] at the specified index.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Insert(
    int index,
    T instance
)
```

VB

```
Public Sub Insert (
    index As Integer,
    instance As T
)
```

Parameters

index	Type: System.Int32 The instance.
instance	Type: T [▶ 1887] The item.

Implements

IList.T..Insert(Int32, T)

Exceptions

Exception	Condition
ArgumentOutOfRangeException	index or index
ArgumentNullException	

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.13 InstanceCollection.T..Remove Method

Removes the specified item.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Remove(
    T item
)
```

VB

```
Public Function Remove (
    item As T
) As Boolean
```

Parameters

item	Type: T [▶ 1887] The item.

Return Value

Type: Boolean
true if item was successfully removed from the ICollection.T.; otherwise, false. This method also returns false if item is not found in the original ICollection.T..

Implements

ICollection.T..Remove(T)

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.14 InstanceCollection.T..RemoveAt Method

Removes the IInstance [▶ 1556] at the specified index.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void RemoveAt(
    int index
)
```

VB

```
Public Sub RemoveAt (
    index As Integer
)
```

Parameters

index	Type: System.Int32 The index.

Implements

IList.T..RemoveAt(Int32)

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.15 InstanceCollection.T..TryGetInstance Method

Tries to get the IInstance [▶ 1556]. of the specified path.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetInstance(
    string instanceSpecifier,
    out T symbol
)
```

VB

```
Public Function TryGetInstance (
    instanceSpecifier As String,
    <OutAttribute> ByRef symbol As T
) As Boolean
```

Parameters

instanceSpecifier	Type: System.String The instance path or Instance Name (dependent of Mode [▶ 1894] setting)
symbol	Type: T [▶ 1887]. The symbol.

Return Value

Type: Boolean
true if the IInstance [▶ 1556] is found; otherwise, false

Implements

IInstanceCollection.T..TryGetInstance(String, T.) [▶ 1884]

Exceptions

Exception	Condition
ArgumentNullException	instancePath
ArgumentException	

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.3.16 InstanceCollection.T..TryGetInstanceByName Method

Tries to get Instnaces by name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public virtual bool TryGetInstanceByName(
    string instanceName,
    out IList<T> instances
)
```

VB

```
Public Overridable Function TryGetInstanceByName (
    instanceName As String,
    <OutAttribute> ByRef instances As IList(Of T)
) As Boolean
```

Parameters

instanceName	Type: System.String Name of the instance.
instances	Type: System.Collections.Generic.IList.T [▶ 1887].. The instances found.

Return Value

Type: Boolean
true if the IInstance [▶ 1556] is found; otherwise, false

Implements

IInstanceCollection.T..TryGetInstanceByName(String, IList.T..) [▶ 1884]

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.4 InstanceCollection.T. Fields

The InstanceCollection.T. [▶ 1887] generic type exposes the following members.

Fields

	Name	Description
	_list [▶ 1909]	The _list
	_pathDict [▶ 1909]	The _path dictionary

	Name	Description
	mode [▶ 1910]	The mode this InstanceCollection.T. [▶ 1887] is working in.

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.4.1 InstanceCollection.T._list Field

The _list

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected List<T> _list
```

VB

```
Protected _list As List(Of T)
```

Field Value

Type: List.T [▶ 1887].

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.4.2 InstanceCollection.T._pathDict Field

The _path dictionary

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected Dictionary<string, T> _pathDict
```

VB

```
Protected _pathDict As Dictionary(Of String, T)
```

Field Value

Type: Dictionary.String, T [▶ 1887].

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.5.4.3 InstanceCollection.T..mode Field

The mode this InstanceCollection.T. [▶ 1887] is working in.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected InstanceCollectionMode mode
```

VB

```
Protected mode As InstanceCollectionMode
```

Field Value

Type: InstanceCollectionMode [▶ 1910]

Reference

InstanceCollection.T. Class [▶ 1887]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.6 InstanceCollectionMode Enumeration

Enum InstanceCollectionMode

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public enum InstanceCollectionMode
```

VB

```
Public Enumeration InstanceCollectionMode
```

Members

	Member name	Value	Description
	Names	0	InstanceCollection{T} is organized with InstanceNames instead of Instance Paths
	Path	1	InstanceCollection{T} is organized with InstancePaths in a flat list

	Member name	Value	Description
	PathHierarchy	2	InstanceCollection{T} is organized with InstancePaths in a Hierarchy (Only Root objects appearing)

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.7 ISymbolCollection.T. Interface

Interface ISymbolCollection

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISymbolCollection<T> : IInstanceCollection<T>,
    IList<T>, ICollection<T>, IEnumerable<T>, IEnumerable
where T : ISymbol
```

VB

```
Public Interface ISymbolCollection(Of T As ISymbol)
    Inherits IInstanceCollection(Of T), IList(Of T), ICollection(Of T),
    IEnumerable(Of T), IEnumerable
```

Type Parameters

T

The ISymbolCollection.T. type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ICollection.T.. (Inherited from ICollection.T..)
	IsReadOnly	Gets a value indicating whether the ICollection.T. is read-only. (Inherited from ICollection.T..)
	Item.String. [▶ 1879]	Gets the IInstance [▶ 1556] with the specified instance path. (Inherited from IInstanceCollection.T. [▶ 1877].)
	Item.Int32.	Gets or sets the element at the specified index. (Inherited from IList.T..)
	Mode [▶ 1880]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from IInstanceCollection.T. [▶ 1877].)

Methods

	Name	Description
	Add	Adds an item to the ICollection.T.. (Inherited from ICollection.T..)
	Clear	Removes all items from the ICollection.T.. (Inherited from ICollection.T..)
	Contains(String) [▶ 1881]	Determines whether this collection contains an instance with the specified instance path. (Inherited from IInstanceCollection.T. [▶ 1877].)

	Name	Description
	Contains(T)	Determines whether the ICollection.T. contains a specific value. (Inherited from ICollection.T..)
	ContainsName [▶ 1882]	Determines whether this collection contains an instance with the specified instance name. (Inherited from IInstanceCollection.T. [▶ 1877].)
	CopyTo	Copies the elements of the ICollection.T. to an Array, starting at a particular Array index. (Inherited from ICollection.T..)
	GetEnumerator	Returns an enumerator that iterates through the collection. (Inherited from IEnumerable.T..)
	GetInstance [▶ 1883]	Gets the IInstance [▶ 1556]by instance path. (Inherited from IInstanceCollection.T. [▶ 1877].)
	GetInstanceByName [▶ 1883]	Gets the IInstance [▶ 1556] by instance name. (Inherited from IInstanceCollection.T. [▶ 1877].)
	IndexOf	Determines the index of a specific item in the IList.T.. (Inherited from IList.T..)
	Insert	Inserts an item to the IList.T. at the specified index. (Inherited from IList.T..)
	Remove	Removes the first occurrence of a specific object from the ICollection.T.. (Inherited from ICollection.T..)
	RemoveAt	Removes the IList.T. item at the specified index. (Inherited from IList.T..)
	TryGetInstance [▶ 1884]	Tries to get the specified instance. (Inherited from IInstanceCollection.T. [▶ 1877].)
	TryGetInstanceByName [▶ 1884]	Tries to get the specified instance by name. (Inherited from IInstanceCollection.T. [▶ 1877].)

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

TwinCAT.TypeSystem.Generic.IInstanceCollection.T. [▶ 1877]

5.9.7.1 ISymbolCollection.T. Properties

The ISymbolCollection.T. [▶ 1911] generic type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ICollection.T.. (Inherited from ICollection.T [▶ 1911]..)
	IsReadOnly	Gets a value indicating whether the ICollection.T. is read-only. (Inherited from ICollection.T [▶ 1911]..)
	Item.String. [▶ 1879]	Gets the IInstance [▶ 1556] with the specified instance path. (Inherited from IInstanceCollection.T. [▶ 1877].)
	Item.Int32.	Gets or sets the element at the specified index. (Inherited from IList.T [▶ 1911]..)
	Mode [▶ 1880]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from IInstanceCollection.T. [▶ 1877].)

Reference

ISymbolCollection.T. Interface [▶ 1911]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.7.2 ISymbolCollection.T. Methods

The ISymbolCollection.T. [▶ 1911] generic type exposes the following members.

Methods

	Name	Description
	Add	Adds an item to the ICollection.T.. (Inherited from ICollection.T [▶ 1911]..)
	Clear	Removes all items from the ICollection.T.. (Inherited from ICollection.T [▶ 1911]..)
	Contains(String) [▶ 1881]	Determines whether this collection contains an instance with the specified instance path. (Inherited from IInstanceCollection.T. [▶ 1877].)
	Contains(T)	Determines whether the ICollection.T. contains a specific value. (Inherited from ICollection.T [▶ 1911]..)
	ContainsName [▶ 1882]	Determines whether this collection contains an instance with the specified instance name. (Inherited from IInstanceCollection.T. [▶ 1877].)
	CopyTo	Copies the elements of the ICollection.T. to an Array, starting at a particular Array index. (Inherited from ICollection.T [▶ 1911]..)
	GetEnumerator	Returns an enumerator that iterates through the collection. (Inherited from IEnumerable.T [▶ 1911]..)
	GetInstance [▶ 1883]	Gets the IInstance [▶ 1556]by instance path. (Inherited from IInstanceCollection.T. [▶ 1877].)
	GetInstanceByName [▶ 1883]	Gets the IInstance [▶ 1556] by instance name. (Inherited from IInstanceCollection.T. [▶ 1877].)
	IndexOf	Determines the index of a specific item in the IList.T.. (Inherited from IList.T [▶ 1911]..)
	Insert	Inserts an item to the IList.T. at the specified index. (Inherited from IList.T [▶ 1911]..)
	Remove	Removes the first occurrence of a specific object from the ICollection.T.. (Inherited from ICollection.T [▶ 1911]..)
	RemoveAt	Removes the IList.T. item at the specified index. (Inherited from IList.T [▶ 1911]..)
	TryGetInstance [▶ 1884]	Tries to get the specified instance. (Inherited from IInstanceCollection.T. [▶ 1877].)
	TryGetInstanceByName [▶ 1884]	Tries to get the specified instance by name. (Inherited from IInstanceCollection.T. [▶ 1877].)

Reference

ISymbolCollection.T. Interface [▶ 1911]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.8 ISymbolProvider.N, T, S. Interface

Symbol provider interface

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public interface ISymbolProvider<N, T, S>
where N : Object, INamespace<T>
where T : IDataType
where S : ISymbol
```

VB

```
Public Interface ISymbolProvider(Of N As {Object, INamespace(Of T)}, T As IDataType, S As ISymbol)
```

Type Parameters

N	Namespace type
T	DataType type
S	Symbol type

The ISymbolProvider.N, T, S. type exposes the following members.

Properties

	Name	Description
	DataTypes [▶ 1915]	Gets all data types from all Namespaces
	Namespaces [▶ 1915]	Get the Namespaces of DataTypes for this Symbol provider
	RootNamespace [▶ 1916]	Gets the root (main) namespace of the Symbol provider.
	RootNamespaceName [▶ 1916]	Gets the name of the root namespace
	Symbols [▶ 1917]	Gets the (root) symbols of the Symbol provider.

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.8.1 ISymbolProvider.N, T, S. Properties

The ISymbolProvider.N, T, S. [▶ 1914] generic type exposes the following members.

Properties

	Name	Description
	DataTypes [▶ 1915]	Gets all data types from all Namespaces
	Namespaces [▶ 1915]	Get the Namespaces of DataTypes for this Symbol provider

	Name	Description
	RootNamespace [▶ 1916]	Gets the root (main) namespace of the Symbol provider.
	RootNamespaceName [▶ 1916]	Gets the name of the root namespace
	Symbols [▶ 1917]	Gets the (root) symbols of the Symbol provider.

Reference

ISymbolProvider.N, T, S. Interface [▶ 1914]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.8.1.1 ISymbolProvider.N, T, S..DataTypes Property

Gets all data types from all Namespaces

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyDataTypeCollection<T> DataTypes { get; }
```

VB

```
ReadOnly Property DataTypes As ReadOnlyDataTypeCollection(Of T)
    Get
```

Property Value

Type: ReadOnlyDataTypeCollection [▶ 1935].T [▶ 1914].
The data types.

Reference

ISymbolProvider.N, T, S. Interface [▶ 1914]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.8.1.2 ISymbolProvider.N, T, S..Namespaces Property

Get the Namespaces of DataTypes for this Symbol provider

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlyNamespaceCollection<N, T> Namespaces { get; }
```

VB

```
ReadOnly Property Namespaces As ReadOnlyNamespaceCollection(Of N, T)
    Get
```

Property Value

Type: ReadOnlyNamespaceCollection [▶ 1953].N [▶ 1914], T [▶ 1914].
ReadOnly collection of the namespaces.

Reference

ISymbolProvider.N, T, S. Interface [▶ 1914]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.8.1.3 ISymbolProvider.N, T, S..RootNamespace Property

Gets the root (main) namespace of the Symbol provider.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
N RootNamespace { get; }
```

VB

```
ReadOnly Property RootNamespace As N
    Get
```

Property Value

Type: N [▶ 1914]
The root namespace.

Reference

ISymbolProvider.N, T, S. Interface [▶ 1914]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.8.1.4 ISymbolProvider.N, T, S..RootNamespaceName Property

Gets the name of the root namespace

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
string RootNamespaceName { get; }
```

VB

```
ReadOnly Property RootNamespaceName As String
    Get
```

Property Value

Type: String
The namespace.

Reference

ISymbolProvider.N, T, S. Interface [▶ 1914]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.8.1.5 ISymbolProvider.N, T, S..Symbols Property

Gets the (root) symbols of the Symbol provider.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
ReadOnlySymbolCollection<S> Symbols { get; }
```

VB

```
ReadOnly Property Symbols As ReadOnlySymbolCollection(Of S)
    Get
```

Property Value

Type: ReadOnlySymbolCollection [▶ 1960].S [▶ 1914].
Read only collection of the Symbols

Reference

ISymbolProvider.N, T, S. Interface [▶ 1914]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9 NamespaceCollection.N, T. Class

Generic class for Namespace collections

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.Generic.NamespaceCollection.N, T.
Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class NamespaceCollection<N, T> : IList<N>,
    ICollection<N>, IEnumerable<N>, IEnumerable, INamespaceCollection<N, T>
where N : Object, INamespace<T>
where T : IDataType
```

VB

```
Public Class NamespaceCollection(Of N As {Object, INamespace(Of T)}, T As IDataType)
    Implements IList(Of N), ICollection(Of N),
    IEnumerable(Of N), IEnumerable, INamespaceCollection(Of N, T)
```

Type Parameters

N T

The NamespaceCollection.N, T. type exposes the following members.

Constructors

	Name	Description
	NamespaceCollection.N, T. [▶ 1919]	Initializes a new instance of the NamespaceCollection.N, T. class.

Properties

	Name	Description
	AllTypes [▶ 1920]	Gets all types included in all namespaces.
	Count [▶ 1920]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1921]	Gets a value indicating whether the ICollection.T. is read-only.
	Item.Int32. [▶ 1922]	Gets or sets the element at the specified index.
	Item.String. [▶ 1923]	Gets or sets the element at the specified index.

Methods

	Name	Description
	Add [▶ 1924]	Adds an item to the ICollection.T..
	Clear [▶ 1925]	Removes all items from the ICollection.T..
	Contains [▶ 1925]	Determines whether the ICollection.T. contains a specific value.
	ContainsNamespace [▶ 1926]	Determines whether the specified name contains namespace.
	CopyTo [▶ 1927]	Copies to.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1927]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1928]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1929]	Inserts an item to the IList.T. at the specified index.

	Name	Description
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1929]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1930]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetNamespace [▶ 1931]	Tries to get the namespace object
	TryGetType [▶ 1931]	Tries to get the specified type.
	TryGetTypeByFullName [▶ 1932]	Tries to get the data type by full name.

Fields

	Name	Description
	allTypes [▶ 1933]	Dictionary FullPath -> IDataType
	list [▶ 1934]	List of Namespaces
	namespaceDict [▶ 1934]	Dictionary NamespaceName --> INamespace
	readOnly [▶ 1934]	Read only indicator

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.1 NamespaceCollection.N, T. Constructor

Initializes a new instance of the NamespaceCollection.N, T. [▶ 1917] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public NamespaceCollection()
```

VB

```
Public Sub New
```

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.2 NamespaceCollection.N, T. Properties

The NamespaceCollection.N, T. [▶ 1917] generic type exposes the following members.

Properties

	Name	Description
	AllTypes [▶ 1920]	Gets all types included in all namespaces.
	Count [▶ 1920]	Gets the number of elements contained in the ICollection.T..
	IsReadOnly [▶ 1921]	Gets a value indicating whether the ICollection.T. is read-only.
	Item.Int32. [▶ 1922]	Gets or sets the element at the specified index.
	Item.String. [▶ 1923]	Gets or sets the element at the specified index.

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.2.1 NamespaceCollection.N, T..AllTypes Property

Gets all types included in all namespaces.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyDataTypeCollection<T> AllTypes { get; }
```

VB

```
Public ReadOnly Property AllTypes As ReadOnlyDataTypeCollection(Of T)
    Get
```

Property Value

Type: ReadOnlyDataTypeCollection [▶ 1935].T [▶ 1917].
All types.

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.2.2 NamespaceCollection.N, T..Count Property

Gets the number of elements contained in the ICollection.T..

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int Count { get; }
```

VB

```
Public ReadOnly Property Count As Integer
    Get
```

Return Value

Type: Int32
The number of elements contained in the ICollection.T..

Implements

ICollection.T..Count

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.2.3 NamespaceCollection.N, T..IsReadOnly Property

Gets a value indicating whether the ICollection.T. is read-only.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool IsReadOnly { get; }
```

VB

```
Public ReadOnly Property IsReadOnly As Boolean
    Get
```

Return Value

Type: Boolean
true if the ICollection.T. is read-only; otherwise, false.

Implements

ICollection.T..IsReadOnly

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.2.4 NamespaceCollection.N, T..Item Property

Overload List

	Name	Description
	Item.Int32. [▶ 1922]	Gets or sets the element at the specified index.
	Item.String. [▶ 1923]	Gets or sets the element at the specified index.

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

NamespaceCollection.N, T..Item Property (Int32)

Gets or sets the element at the specified index.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public N this[
    int index
] { get; set; }
```

VB

```
Public Default Property Item (
    index As Integer
) As N
    Get
    Set
```

Parameters

index	Type: System.Int32 The index.

Return Value

Type: N [▶ 1917]

Implements

IList.T..Item.Int32.

Exceptions

Exception	Condition
NotImplementedException	

Reference

NamespaceCollection.N, T. Class [▶ 1917]

Item Overload [▶ 1922]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

NamespaceCollection.N, T..Item Property (String)

Gets or sets the element at the specified index.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public N this[
    string str
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    str As String
) As N
    Get
```

Parameters

str	Type: System.String The STR.

Return Value

Type: N [▶ 1917]

Reference

NamespaceCollection.N, T. Class [▶ 1917]

Item Overload [▶ 1922]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3 NamespaceCollection.N, T. Methods

The NamespaceCollection.N, T. [▶ 1917] generic type exposes the following members.

Methods

	Name	Description
	Add [▶ 1924]	Adds an item to the ICollection.T..
	Clear [▶ 1925]	Removes all items from the ICollection.T..
	Contains [▶ 1925]	Determines whether the ICollection.T. contains a specific value.

	Name	Description
	ContainsNamespace [▶ 1926]	Determines whether the specified name contains namespace.
	CopyTo [▶ 1927]	Copies to.
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1927]	Returns an enumerator that iterates through the collection.
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1928]	Determines the index of a specific item in the IList.T..
	Insert [▶ 1929]	Inserts an item to the IList.T. at the specified index.
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1929]	Removes the first occurrence of a specific object from the ICollection.T..
	RemoveAt [▶ 1930]	Removes the IList.T. item at the specified index.
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetNamespace [▶ 1931]	Tries to get the namespace object
	TryGetType [▶ 1931]	Tries to get the specified type.
	TryGetTypeByFullName [▶ 1932]	Tries to get the data type by full name.

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.1 NamespaceCollection.N, T..Add Method

Adds an item to the ICollection.T..

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Add(
    N item
)
```

VB

```
Public Sub Add (
    item As N
)
```

Parameters

item	Type: N [▶ 1917] The object to add to the ICollection.T..

Implements

ICollection.T..Add(T)

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.2 NamespaceCollection.N, T..Clear Method

Removes all items from the ICollection.T..

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Clear()
```

VB

```
Public Sub Clear
```

Implements

ICollection.T..Clear.

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.3 NamespaceCollection.N, T..Contains Method

Determines whether the ICollection.T. contains a specific value.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    N item
)
```

VB

```
Public Function Contains (
    item As N
) As Boolean
```

Parameters

item	Type: N [▶ 1917] The object to locate in the ICollection.T..

Return Value

Type: Boolean
true if item is found in the ICollection.T.; otherwise, false.

Implements

ICollection.T..Contains(T)

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.4 NamespaceCollection.N, T..ContainsNamespace Method

Determines whether the specified name contains namespace.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool ContainsNamespace(
    string name
)
```

VB

```
Public Function ContainsNamespace (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name.

Return Value

Type: Boolean
true if the specified name contains namespace; otherwise, false.

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.5 NamespaceCollection.N, T..CopyTo Method

Copies to.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void CopyTo(
    N[] array,
    int arrayIndex
)
```

VB

```
Public Sub CopyTo (
    array As N(),
    arrayIndex As Integer
)
```

Parameters

array	Type: .N [▶ 1917]. The array.
arrayIndex	Type: System.Int32 Index of the array.

Implements

ICollection.T..CopyTo(.T., Int32)

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.6 NamespaceCollection.N, T..GetEnumerator Method

Returns an enumerator that iterates through the collection.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumerator<N> GetEnumerator()
```

VB

```
Public Function GetEnumerator As IEnumerator(Of N)
```

Return Value

Type: IEnumerator.N [▶ 1917].
A IEnumerator.T. that can be used to iterate through the collection.

Implements

IEnumerable.T..GetEnumerator.

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.7 NamespaceCollection.N, T..IndexOf Method

Determines the index of a specific item in the IList.T..

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public int IndexOf(
    N item
)
```

VB

```
Public Function IndexOf (
    item As N
) As Integer
```

Parameters

item	Type: N [▶ 1917] The object to locate in the IList.T..

Return Value

Type: Int32
The index of item if found in the list; otherwise, -1.

Implements

IList.T..IndexOf(T)

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.8 NamespaceCollection.N, T..Insert Method

Inserts an item to the IList.T. at the specified index.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void Insert(
    int index,
    N item
)
```

VB

```
Public Sub Insert (
    index As Integer,
    item As N
)
```

Parameters

index	Type: System.Int32 The zero-based index at which item should be inserted.
item	Type: N [▶ 1917] The object to insert into the IList.T..

Implements

IList.T..Insert(Int32, T)

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.9 NamespaceCollection.N, T..Remove Method

Removes the first occurrence of a specific object from the ICollection.T..

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Remove(
    N item
)
```

VB

```
Public Function Remove (
    item As N
) As Boolean
```

Parameters

item	Type: N [▶ 1917] The object to remove from the ICollection.T..

Return Value

Type: Boolean
true if item was successfully removed from the ICollection.T.; otherwise, false. This method also returns false if item is not found in the original ICollection.T..

Implements

ICollection.T..Remove(T)

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.10 NamespaceCollection.N, T..RemoveAt Method

Removes the IList.T. item at the specified index.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public void RemoveAt(
    int index
)
```

VB

```
Public Sub RemoveAt (
    index As Integer
)
```

Parameters

index	Type: System.Int32 The zero-based index of the item to remove.

Implements

IList.T..RemoveAt(Int32)

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.11 NamespaceCollection.N, T..TryGetNamespace Method

Tries to get the namespace object

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetNamespace(
    string name,
    out N nspace
)
```

VB

```
Public Function TryGetNamespace (
    name As String,
    <OutAttribute> ByRef nspace As N
) As Boolean
```

Parameters

name	Type: System.String The name.
nspace	Type: N [▶ 1917]. The namespace object (out-parameter)

Return Value

Type: Boolean
true if found, false if not contained.

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.12 NamespaceCollection.N, T..TryGetType Method

Tries to get the specified type.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetType(
    string typeName,
    out T dataType
)
```

VB

```
Public Function TryGetType (
    typeName As String,
    <OutAttribute> ByRef dataType As T
) As Boolean
```

Parameters

typeName	Type: System.String Data type name
dataType	Type: T [▶ 1917]. The found data type (out-parameter).

Return Value

Type: Boolean
true if found, false if not contained.

Exceptions

Exception	Condition
ArgumentNullException	typeName
ArgumentException	

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.3.13 NamespaceCollection.N, T..TryGetTypeByFullName Method

Tries to get the data type by full name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetTypeByFullName(
    string fullname,
    out T dataType
)
```

VB

```
Public Function TryGetTypeByFullName (
    fullname As String,
    <OutAttribute> ByRef dataType As T
) As Boolean
```

Parameters

fullname	Type: System.String DataTypes full name.
dataType	Type: T [▶ 1917]. Found data type (out-parameter).

Return Value

Type: Boolean
true if found, false if not contained.

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.4 NamespaceCollection.N, T. Fields

The NamespaceCollection.N, T. [▶ 1917] generic type exposes the following members.

Fields

	Name	Description
	allTypes [▶ 1933]	Dictionary FullPath -> IDataType
	list [▶ 1934]	List of Namespaces
	namespaceDict [▶ 1934]	Dictionary NamespaceName --> INamespace
	readOnly [▶ 1934]	Read only indicator

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.4.1 NamespaceCollection.N, T..allTypes Field

Dictionary FullPath -> IDataType

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected Dictionary<string, T> allTypes
```

VB

```
Protected allTypes As Dictionary(Of String, T)
```

Field Value

Type: Dictionary.String, T [▶ 1917].

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.4.2 NamespaceCollection.N, T..list Field

List of Namespaces

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected List<N> list
```

VB

```
Protected list As List(Of N)
```

Field Value

Type: List.N [▶ 1917].

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.4.3 NamespaceCollection.N, T..namespaceDict Field

Dictionary NamespaceName --> INamespace

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected Dictionary<string, N> namespaceDict
```

VB

```
Protected namespaceDict As Dictionary(Of String, N)
```

Field Value

Type: Dictionary.String, N [▶ 1917].

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.9.4.4 NamespaceCollection.N, T..readOnly Field

Read only indicator

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected bool readOnly
```

VB

```
Protected readOnly As Boolean
```

Field Value

Type: Boolean

Reference

NamespaceCollection.N, T. Class [▶ 1917]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.10 ReadOnlyDataTypeCollection.T. Class

ReadOnly DataType collection

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.T.
    TwinCAT.TypeSystem.Generic.ReadOnlyDataTypeCollection.T.
      TwinCAT.TypeSystem.ReadOnlyDataTypeCollection [▶ 1727]

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyDataTypeCollection<T> : ReadOnlyCollection<T>,
    IDataTypeContainer<T>
where T : IDataType
```

VB

```
Public Class ReadOnlyDataTypeCollection(Of T As IDataType)
    Inherits ReadOnlyCollection(Of T)
    Implements IDataTypeContainer(Of T)
```

Type Parameters

T

The ReadOnlyDataTypeCollection.T. type exposes the following members.

Constructors

	Name	Description
	ReadOnlyDataTypeCollection.T. (DataTypeCollection.T.) [▶ 1937]	Initializes a new instance of the ReadOnlyDataTypeCollection.T. class.

	Name	Description
	ReadOnlyDataTypeCollection.T.(ReadOnlyDataTypeCollection.T.) [▶ 1938]	Initializes a new instance of the ReadOnlyDataTypeCollection.T. class.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.T..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.T..)
	Item.String. [▶ 1939]	Gets the element with the specified type name.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.T..)

Methods

	Name	Description
	Contains	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T..)
	ContainsType [▶ 1940]	Determines whether the specified name contains type.
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.T..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetType [▶ 1941]	Tries to get the Type with the specified name out of the collection.

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.10.1 ReadOnlyDataTypeCollection.T. Constructor

Overload List

	Name	Description
	ReadOnlyDataTypeCollection.T. (DataTypeCollection.T.) [▶ 1937]	Initializes a new instance of the ReadOnlyDataTypeCollection.T. [▶ 1935] class.
	ReadOnlyDataTypeCollection.T. (ReadOnlyDataTypeCollection.T.) [▶ 1938]	Initializes a new instance of the ReadOnlyDataTypeCollection.T. [▶ 1935] class.

Reference

ReadOnlyDataTypeCollection.T. Class [▶ 1935]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.10.1.1 ReadOnlyDataTypeCollection.T. Constructor (DataTypeCollection.T.)

Initializes a new instance of the ReadOnlyDataTypeCollection.T. [▶ 1935] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyDataTypeCollection(
    DataTypeCollection<T> coll
)
```

VB

```
Public Sub New (
    coll As DataTypeCollection(Of T)
)
```

Parameters

coll	Type: TwinCAT.TypeSystem.Generic.DataTypeCollection [▶ 1855].T [▶ 1935]. The collection.

Reference

ReadOnlyDataTypeCollection.T. Class [▶ 1935]

ReadOnlyDataTypeCollection.T. Overload [▶ 1937]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.10.1.2 ReadOnlyDataTypeCollection.T. Constructor (ReadOnlyDataTypeCollection.T.)

Initializes a new instance of the ReadOnlyDataTypeCollection.T. [▶ 1935] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyDataTypeCollection(
    ReadOnlyDataTypeCollection<T> coll
)
```

VB

```
Public Sub New (
    coll As ReadOnlyDataTypeCollection(Of T)
)
```

Parameters

coll	Type: TwinCAT.TypeSystem.Generic.ReadOnlyDataTypeCollection [▶ 1935].T [▶ 1935]. The coll.

Reference

ReadOnlyDataTypeCollection.T. Class [▶ 1935]

ReadOnlyDataTypeCollection.T. Overload [▶ 1937]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.10.2 ReadOnlyDataTypeCollection.T. Properties

The ReadOnlyDataTypeCollection.T. [▶ 1935] generic type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.T [▶ 1935]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.T [▶ 1935]..)
	Item.String. [▶ 1939]	Gets the element with the specified type name.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.T [▶ 1935]..)

Reference

ReadOnlyDataTypeCollection.T. Class [▶ 1935]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.10.2.1 ReadOnlyDataTypeCollection.T..Item Property

Overload List

	Name	Description
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.T [▶ 1935]..)
	Item.String. [▶ 1939]	Gets the element with the specified type name.

Reference

ReadOnlyDataTypeCollection.T. Class [▶ 1935]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

ReadOnlyDataTypeCollection.T..Item Property (String)

Gets the element with the specified type name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T this[
    string name
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    name As String
) As T
    Get
```

Parameters

name	Type: System.String The name.

Return Value

Type: T [▶ 1935]
T.

Implements

IDataTypeContainer.T..Item.String. [▶ 1875]

Reference

ReadOnlyDataTypeCollection.T. Class [▶ 1935]

Item Overload [▶ 1939]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.10.3 ReadOnlyDataTypeCollection.T. Methods

The ReadOnlyDataTypeCollection.T. [▶ 1935] generic type exposes the following members.

Methods

	Name	Description
	Contains	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T [▶ 1935]..)
	ContainsType [▶ 1940]	Determines whether the specified name contains type.
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.T [▶ 1935]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T [▶ 1935]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T [▶ 1935]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetType [▶ 1941]	Tries to get the Type with the specified name out of the collection.

Reference

ReadOnlyDataTypeCollection.T. Class [▶ 1935]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.10.3.1 ReadOnlyDataTypeCollection.T..ContainsType Method

Determines whether the specified name contains type.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool ContainsType(
    string name
)
```

VB

```
Public Function ContainsType (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name.

Return Value

Type: Boolean
true if the specified name contains type; otherwise, false.

Implements

IDataTypeContainer.T..ContainsType(String) [▶ 1875]

Reference

ReadOnlyDataTypeCollection.T. Class [▶ 1935]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.10.3.2 ReadOnlyDataTypeCollection.T..TryGetType Method

Tries to get the Type with the specified name out of the collection.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetType(
    string name,
    out T type
)
```

VB

```
Public Function TryGetType (
    name As String,
    <OutAttribute> ByRef type As T
) As Boolean
```

Parameters

name	Type: System.String The name.
type	Type: T [▶ 1935]. The type.

Return Value

Type: Boolean
true if found

Implements

IDataTypeContainer.T..TryGetType(String, T.) [▶ 1876]

Reference

ReadOnlyDataTypeCollection.T. Class [▶ 1935]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11 ReadOnlyInstanceCollection.T. Class

ReadOnly Instance collection

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.T.
    TwinCAT.TypeSystem.Generic.ReadOnlyInstanceCollection.T.
      TwinCAT.TypeSystem.Generic.ReadOnlySymbolCollection.T. [▶ 1960]
      TwinCAT.TypeSystem.ReadOnlyFieldCollection [▶ 1750]
      TwinCAT.TypeSystem.ReadOnlyMemberCollection [▶ 1755]

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyInstanceCollection<T> : ReadOnlyCollection<T>,
    IInstanceCollection<T>, IList<T>, ICollection<T>, IEnumerable<T>,
    IEnumerable
where T : IInstance
```

VB

```
Public Class ReadOnlyInstanceCollection(Of T As IInstance)
    Inherits ReadOnlyCollection(Of T)
    Implements IInstanceCollection(Of T), IList(Of T),
    ICollection(Of T), IEnumerable(Of T), IEnumerable
```

Type Parameters

T

The ReadOnlyInstanceCollection.T. type exposes the following members.

Constructors

	Name	Description
	ReadOnlyInstanceCollection.T. [▶ 1944]	Initializes a new instance of the ReadOnlyInstanceCollection.T. class.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.T..)

	Name	Description
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.T..)
	Item.String. [▶ 1945]	Gets the element with the specified instance path.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.T..)
	Mode [▶ 1946]	Gets the InstanceCollectionMode [▶ 1910].

Methods

	Name	Description
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T..)
	Contains(String) [▶ 1947]	Determines whether the ReadOnlyInstanceCollection.T. contains an instance with the specified instance path.
	ContainsName [▶ 1948]	Determines whether the specified instance is contained.
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.T..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1949]	Gets the IInstance [▶ 1556]by instance path.
	GetInstanceByName [▶ 1950]	Gets the IInstance [▶ 1556] by instance name.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1950]	Tries to get the instance with the specified instance path.
	TryGetInstanceByName [▶ 1951]	Tries to get the instance by name.

Fields

	Name	Description
	mode [▶ 1952]	Mode of the IInstanceCollection.T. [▶ 1877]

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.1 ReadOnlyInstanceCollection.T. Constructor

Initializes a new instance of the ReadOnlyInstanceCollection.T. [▶ 1942] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyInstanceCollection(
    IInstanceCollection<T> coll
)
```

VB

```
Public Sub New (
    coll As IInstanceCollection(Of T)
)
```

Parameters

coll	Type: TwinCAT.TypeSystem.Generic.IInstanceCollection [▶ 1877].T [▶ 1942]. The coll.

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.2 ReadOnlyInstanceCollection.T. Properties

The ReadOnlyInstanceCollection.T. [▶ 1942] generic type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.T [▶ 1942]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.T [▶ 1942]..)
	Item.String. [▶ 1945]	Gets the element with the specified instance path.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.T [▶ 1942]..)
	Mode [▶ 1946]	Gets the InstanceCollectionMode [▶ 1910].

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.2.1 ReadOnlyInstanceCollection.T..Item Property

Overload List

	Name	Description
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.T [▶ 1942]..)
	Item.String. [▶ 1945]	Gets the element with the specified instance path.

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

ReadOnlyInstanceCollection.T..Item Property (String)

Gets the element with the specified instance path.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T this[
    string instancePath
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    instancePath As String
) As T
    Get
```

Parameters

instancePath	Type: System.String The instance path.

Return Value

Type: T [▶ 1942]
The instance if contained.

Implements

IInstanceCollection.T..Item.String. [▶ 1879]

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

Item Overload [▶ 1945]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.2.2 ReadOnlyInstanceCollection.T..Mode Property

Gets the InstanceCollectionMode [▶ 1910].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public InstanceCollectionMode Mode { get; }
```

VB

```
Public ReadOnly Property Mode As InstanceCollectionMode
    Get
```

Property Value

Type: InstanceCollectionMode [▶ 1910]
The mode.

Implements

IInstanceCollection.T..Mode [▶ 1880]

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.3 ReadOnlyInstanceCollection.T. Methods

The ReadOnlyInstanceCollection.T. [▶ 1942] generic type exposes the following members.

Methods

	Name	Description
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T [▶ 1942]..)
	Contains(String) [▶ 1947]	Determines whether the ReadOnlyInstanceCollection.T. [▶ 1942] contains an instance with the specified instance path.
	ContainsName [▶ 1948]	Determines whether the specified instance is contained.
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.T [▶ 1942]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T [▶ 1942]..)

	Name	Description
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1949]	Gets the IInstance [▶ 1556]by instance path.
	GetInstanceByName [▶ 1950]	Gets the IInstance [▶ 1556] by instance name.
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T [▶ 1942]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1950]	Tries to get the instance with the specified instance path.
	TryGetInstanceByName [▶ 1951]	Tries to get the instance by name.

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.3.1 ReadOnlyInstanceCollection.T..Contains Method

Overload List

	Name	Description
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T [▶ 1942]..)
	Contains(String) [▶ 1947]	Determines whether the ReadOnlyInstanceCollection.T. [▶ 1942] contains an instance with the specified instance path.

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

ReadOnlyInstanceCollection.T..Contains Method (String)

Determines whether the ReadOnlyInstanceCollection.T. [▶ 1942] contains an instance with the specified instance path.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool Contains(
    string instancePath
)
```

VB

```
Public Function Contains (
    instancePath As String
) As Boolean
```

Parameters

instancePath	Type: System.String The instance path.

Return Value

Type: Boolean
true if contains the specified instance path; otherwise, false.

Implements

IInstanceCollection.T..Contains(String) [▶ 1881]

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

Contains Overload [▶ 1947]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.3.2 ReadOnlyInstanceCollection.T..ContainsName Method

Determines whether the specified instance is contained.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool ContainsName(
    string instanceName
)
```

VB

```
Public Function ContainsName (
    instanceName As String
) As Boolean
```

Parameters

instanceName	Type: System.String Name of the instance.

Return Value

Type: Boolean
true, if instance name is found.

Implements

IInstanceCollection.T..ContainsName(String) [▶ 1882]

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.3.3 ReadOnlyInstanceCollection.T..GetInstance Method

Gets the IInstance [▶ 1556]by instance path.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public T GetInstance(
    string instancePath
)
```

VB

```
Public Function GetInstance (
    instancePath As String
) As T
```

Parameters

instancePath	Type: System.String The instance path.

Return Value

Type: T [▶ 1942]
T.

Implements

IInstanceCollection.T..GetInstance(String) [▶ 1883]

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.3.4 ReadOnlyInstanceCollection.T..GetInstanceByName Method

Gets the IInstance [▶ 1556] by instance name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IList<T> GetInstanceByName(
    string instanceName
)
```

VB

```
Public Function GetInstanceByName (
    instanceName As String
) As IList(Of T)
```

Parameters

instanceName	Type: System.String Name of the instance.

Return Value

Type: IList.T [▶ 1942].
IList<T>.

Implements

IInstanceCollection.T..GetInstanceByName(String) [▶ 1883]

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.3.5 ReadOnlyInstanceCollection.T..TryGetInstance Method

Tries to get the instance with the specified instance path.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetInstance(
    string instancePath,
    out T instance
)
```

VB

```
Public Function TryGetInstance (
    instancePath As String,
    <OutAttribute> ByRef instance As T
) As Boolean
```

Parameters

instancePath	Type: System.String The instance path.
instance	Type: T [▶ 1942]. The instance.

Return Value

Type: Boolean
true, if found, false if not contained.

Implements

IInstanceCollection.T..TryGetInstance(String, T.) [▶ 1884]

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.3.6 ReadOnlyInstanceCollection.T..TryGetInstanceByName Method

Tries to get the instance by name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetInstanceByName(
    string instanceName,
    out IList<T> symbols
)
```

VB

```
Public Function TryGetInstanceByName (
    instanceName As String,
    <OutAttribute> ByRef symbols As IList(Of T)
) As Boolean
```

Parameters

instanceName	Type: System.String Name of the instance.
symbols	Type: System.Collections.Generic.IList.T [▶ 1942].. The found symbols (out-parameter)

Return Value

Type: Boolean
true, if found; false if not contained.

Implements

IInstanceCollection.T..TryGetInstanceByName(String, IList.T..) [▶ 1884]

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.4 ReadOnlyInstanceCollection.T. Fields

The ReadOnlyInstanceCollection.T. [▶ 1942] generic type exposes the following members.

Fields

	Name	Description
	mode [▶ 1952]	Mode of the IInstanceCollection.T. [▶ 1877]

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.11.4.1 ReadOnlyInstanceCollection.T..mode Field

Mode of the IInstanceCollection.T. [▶ 1877]

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
protected InstanceCollectionMode mode
```

VB

```
Protected mode As InstanceCollectionMode
```

Field Value

Type: InstanceCollectionMode [▶ 1910]

Reference

ReadOnlyInstanceCollection.T. Class [▶ 1942]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.12 ReadOnlyNamespaceCollection.N, T. Class

Read Only namespace collection

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.N.
    TwinCAT.TypeSystem.Generic.ReadOnlyNamespaceCollection.N, T.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlyNamespaceCollection<N, T> : ReadOnlyCollection<N>,
    INamespaceCollection<N, T>
where N : Object, INamespace<T>
where T : IDataType
```

VB

```
Public Class ReadOnlyNamespaceCollection(Of N As {Object, INamespace(Of T)}, T As IDataType)
    Inherits ReadOnlyCollection(Of N)
    Implements INamespaceCollection(Of N, T)
```

Type Parameters

N T

The ReadOnlyNamespaceCollection.N, T. type exposes the following members.

Constructors

	Name	Description
	ReadOnlyNamespaceCollection.N, T. [▶ 1954]	Initializes a new instance of the ReadOnlyNamespaceCollection.N, T. class.

Properties

	Name	Description
	AllTypes [▶ 1955]	Gets all types included in all namespaces.
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.N..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.N..)
	Item.String. [▶ 1956]	Gets the element at the specified index.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.N..)

Methods

	Name	Description
	Contains	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.N..)
	ContainsNamespace [▶ 1958]	Determines whether this collection contains a namespace with the specified name.
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.N..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.N..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.N..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetNamespace [▶ 1958]	Tries to get the namespace with the specified name.
	TryGetType [▶ 1959]	Tries to get the specified data type.
	TryGetTypeByFullName [▶ 1960]	Tries to get the specified type (by fullName)

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.12.1 ReadOnlyNamespaceCollection.N, T. Constructor

Initializes a new instance of the ReadOnlyNamespaceCollection.N, T. [▶ 1953] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyNamespaceCollection(
    NamespaceCollection<N, T> coll
)
```

VB

```
Public Sub New (
    coll As NamespaceCollection(Of N, T)
)
```

Parameters

coll	Type: TwinCAT.TypeSystem.Generic.NamespaceCollection [▶ 1917].N [▶ 1953], T [▶ 1953]. The coll.

Reference

ReadOnlyNamespaceCollection.N, T. Class [▶ 1953]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.12.2 ReadOnlyNamespaceCollection.N, T. Properties

The ReadOnlyNamespaceCollection.N, T. [▶ 1953] generic type exposes the following members.

Properties

	Name	Description
	AllTypes [▶ 1955]	Gets all types included in all namespaces.
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.N [▶ 1953]..)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.N [▶ 1953]..)
	Item.String. [▶ 1956]	Gets the element at the specified index.
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.N [▶ 1953]..)

Reference

ReadOnlyNamespaceCollection.N, T. Class [▶ 1953]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.12.2.1 ReadOnlyNamespaceCollection.N, T..AllTypes Property

Gets all types included in all namespaces.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlyDataTypeCollection<T> AllTypes { get; }
```

VB

```
Public ReadOnly Property AllTypes As ReadOnlyDataTypeCollection(Of T)
    Get
```

Property Value

Type: ReadOnlyDataTypeCollection [▶ 1935].T [▶ 1953].
All types.

Reference

ReadOnlyNamespaceCollection.N, T. Class [▶ 1953]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.12.2.2 ReadOnlyNamespaceCollection.N, T..Item Property

Overload List

	Name	Description
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.N [▶ 1953]..)
	Item.String. [▶ 1956]	Gets the element at the specified index.

Reference

ReadOnlyNamespaceCollection.N, T. Class [▶ 1953]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

ReadOnlyNamespaceCollection.N, T..Item Property (String)

Gets the element at the specified index.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public N this[
    string name
] { get; }
```

VB

```
Public ReadOnly Default Property Item (
    name As String
) As N
    Get
```

Parameters

name	Type: System.String The name.

Return Value

Type: N [▶ 1953]

Reference

ReadOnlyNamespaceCollection.N, T. Class [▶ 1953]

Item Overload [▶ 1956]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.12.3 ReadOnlyNamespaceCollection.N, T. Methods

The ReadOnlyNamespaceCollection.N, T. [▶ 1953] generic type exposes the following members.

Methods

	Name	Description
	Contains	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.N [▶ 1953]..)
	ContainsNamespace [▶ 1958]	Determines whether this collection contains a namespace with the specified name.
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.N [▶ 1953]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.N [▶ 1953]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.N [▶ 1953]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetNamespace [▶ 1958]	Tries to get the namespace with the specified name.
	TryGetType [▶ 1959]	Tries to get the specified data type.
	TryGetTypeByFullName [▶ 1960]	Tries to get the specified type (by fullName)

Reference

ReadOnlyNamespaceCollection.N, T. Class [▶ 1953]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.12.3.1 ReadOnlyNamespaceCollection.N, T..ContainsNamespace Method

Determines whether this collection contains a namespace with the specified name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool ContainsNamespace(
    string name
)
```

VB

```
Public Function ContainsNamespace (
    name As String
) As Boolean
```

Parameters

name	Type: System.String The name of the namespace

Return Value

Type: Boolean
true if the namespace is contained; otherwise, false.

Reference

ReadOnlyNamespaceCollection.N, T. Class [▶ 1953]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.12.3.2 ReadOnlyNamespaceCollection.N, T..TryGetNamespace Method

Tries to get the namespace with the specified name.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetNamespace(
    string name,
    out N nspace
)
```

VB

```
Public Function TryGetNamespace (
    name As String,
    <OutAttribute> ByRef nspace As N
) As Boolean
```

Parameters

name	Type: System.String Namespace name.
nspace	Type: N [▶ 1953]. The found namespace (out-parameter).

Return Value

Type: Boolean
true if found, false if not contained.

Reference

ReadOnlyNamespaceCollection.N, T. Class [▶ 1953]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.12.3.3 ReadOnlyNamespaceCollection.N, T..TryGetType Method

Tries to get the specified data type.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetType(
    string typeName,
    out T dataType
)
```

VB

```
Public Function TryGetType (
    typeName As String,
    <OutAttribute> ByRef dataType As T
) As Boolean
```

Parameters

typeName	Type: System.String Name of the type.
dataType	Type: T [▶ 1953]. Data Type (out-parameter).

Return Value

Type: Boolean
true if found, false if not contained.

Reference

ReadOnlyNamespaceCollection.N, T. Class [▶ 1953]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.12.3.4 ReadOnlyNamespaceCollection.N, T..TryGetTypeByFullName Method

Tries to get the specified type (by fullName)

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetTypeByFullName(
    string fullname,
    out T dataType
)
```

VB

```
Public Function TryGetTypeByFullName (
    fullname As String,
    <OutAttribute> ByRef dataType As T
) As Boolean
```

Parameters

fullname	Type: System.String FullName of the data type.
dataType	Type: T [▶ 1953]. Found Data type (out-parameter).

Return Value

Type: Boolean
true if found, false if not contained.

Reference

ReadOnlyNamespaceCollection.N, T. Class [▶ 1953]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.13 ReadOnlySymbolCollection.T. Class

Read only symbol collection.

Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.ReadOnlyCollection.T.
    TwinCAT.TypeSystem.Generic.ReadOnlyInstanceCollection [▶ 1942].T.
      TwinCAT.TypeSystem.Generic.ReadOnlySymbolCollection.T.
        TwinCAT.TypeSystem.ReadOnlySymbolCollection [▶ 1772]

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class ReadOnlySymbolCollection<T> : ReadOnlyInstanceCollection<T>
where T : ISymbol
```

VB

```
Public Class ReadOnlySymbolCollection(Of T As ISymbol)
    Inherits ReadOnlyInstanceCollection(Of T)
```

Type Parameters

T

The ReadOnlySymbolCollection.T. type exposes the following members.

Constructors

	Name	Description
	ReadOnlySymbolCollection.T. [▶ 1962]	Initializes a new instance of the ReadOnlySymbolCollection.T. class.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.T..)
	Item.String. [▶ 1945]	Gets the element with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.T..)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.T..)
	Mode [▶ 1946]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Methods

	Name	Description
	Contains(String) [▶ 1947]	Determines whether the ReadOnlyInstanceCollection.T. [▶ 1942] contains an instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T..)
	ContainsName [▶ 1948]	Determines whether the specified instance is contained. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.T..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)

	Name	Description
	GetInstance [▶ 1949]	Gets the IInstance [▶ 1556]by instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetInstanceByName [▶ 1950]	Gets the IInstance [▶ 1556] by instance name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1950]	Tries to get the instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	TryGetInstanceByName [▶ 1951]	Tries to get the instance by name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Fields

	Name	Description
	mode [▶ 1952]	Mode of the IInstanceCollection.T. [▶ 1877] (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.13.1 ReadOnlySymbolCollection.T. Constructor

Initializes a new instance of the ReadOnlySymbolCollection.T. [▶ 1960] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlySymbolCollection(
    IInstanceCollection<T> coll
)
```

VB

```
Public Sub New (
    coll As IInstanceCollection(Of T)
)
```

Parameters

coll	Type: TwinCAT.TypeSystem.Generic.IInstanceCollection [▶ 1877].T [▶ 1960]. The coll.

Reference

ReadOnlySymbolCollection.T. Class [▶ 1960]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.13.2 ReadOnlySymbolCollection.T. Properties

The ReadOnlySymbolCollection.T. [▶ 1960] generic type exposes the following members.

Properties

	Name	Description
	Count	Gets the number of elements contained in the ReadOnlyCollection.T. instance. (Inherited from ReadOnlyCollection.T [▶ 1960]..)
	Item.String. [▶ 1945]	Gets the element with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Item.Int32.	Gets the element at the specified index. (Inherited from ReadOnlyCollection.T [▶ 1960]..)
	Items	Returns the IList.T. that the ReadOnlyCollection.T. wraps. (Inherited from ReadOnlyCollection.T [▶ 1960]..)
	Mode [▶ 1946]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

ReadOnlySymbolCollection.T. Class [▶ 1960]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.13.3 ReadOnlySymbolCollection.T. Methods

The ReadOnlySymbolCollection.T. [▶ 1960] generic type exposes the following members.

Methods

	Name	Description
	Contains(String) [▶ 1947]	Determines whether the ReadOnlyInstanceCollection.T. [▶ 1942] contains an instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	Contains(T)	Determines whether an element is in the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T [▶ 1960]..)
	ContainsName [▶ 1948]	Determines whether the specified instance is contained. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	CopyTo	Copies the entire ReadOnlyCollection.T. to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from ReadOnlyCollection.T [▶ 1960]..)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	GetEnumerator	Returns an enumerator that iterates through the ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T [▶ 1960]..)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1949]	Gets the IInstance [▶ 1556]by instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetInstanceByName [▶ 1950]	Gets the IInstance [▶ 1556] by instance name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf	Searches for the specified object and returns the zero-based index of the first occurrence within the entire ReadOnlyCollection.T.. (Inherited from ReadOnlyCollection.T [▶ 1960]..)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1950]	Tries to get the instance with the specified instance path. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)
	TryGetInstanceByName [▶ 1951]	Tries to get the instance by name. (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

ReadOnlySymbolCollection.T. Class [▶ 1960]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.13.4 ReadOnlySymbolCollection.T. Fields

The ReadOnlySymbolCollection.T. [▶ 1960] generic type exposes the following members.

Fields

	Name	Description
	mode [▶ 1952]	Mode of the IInstanceCollection.T. [▶ 1877] (Inherited from ReadOnlyInstanceCollection.T. [▶ 1942].)

Reference

ReadOnlySymbolCollection.T. Class [▶ 1960]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.14 SymbolCollection.T. Class

Interface represents a collection of ISymbol [▶ 1634] objects.

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.Generic.InstanceCollection [▶ 1887].T.
    TwinCAT.TypeSystem.Generic.SymbolCollection.T.

TwinCAT.TypeSystem.SymbolCollection [▶ 1818]
Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SymbolCollection<T> : InstanceCollection<T>,
    ISymbolCollection<T>, IInstanceCollection<T>, IList<T>, ICollection<T>,
    IEnumerable<T>, IEnumerable
where T : class, ISymbol
```

VB

```
Public Class SymbolCollection(Of T As {Class, ISymbol})
    Inherits InstanceCollection(Of T)
    Implements ISymbolCollection(Of T), IInstanceCollection(Of T),
    IList(Of T), ICollection(Of T), IEnumerable(Of T),
    IEnumerable
```

Type Parameters

T

The SymbolCollection.T. type exposes the following members.

Properties

	Name	Description
	Count [▶ 1891]	Gets the collection count. (Inherited from InstanceCollection.T. [▶ 1887].)
	IsReadOnly [▶ 1892]	Gets a value indicating whether this instance is read only. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.Int32. [▶ 1893]	Gets or sets the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.String. [▶ 1894]	Gets the IInstance [▶ 1556] with the specified instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	Mode [▶ 1894]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from InstanceCollection.T. [▶ 1887].)

Methods

	Name	Description
	Add [▶ 1896]	Adds the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	AddRange [▶ 1897]	Adds the specified items to this collection. (Inherited from InstanceCollection.T. [▶ 1887].)
	AsReadOnly [▶ 1968]	Returns a Read only version of this collection (shallow copy).
	Clear [▶ 1898]	Clears this instance. (Inherited from InstanceCollection.T. [▶ 1887].)
	Clone [▶ 1969]	Clones this instance.
	Contains(String) [▶ 1899]	Determines whether this collection contains an IInstance [▶ 1556] with the specified InstanceName / InstancePath (Inherited from InstanceCollection.T. [▶ 1887].)
	Contains(T) [▶ 1899]	Determines whether this collection contains the specified IInstance [▶ 1556] (Inherited from InstanceCollection.T. [▶ 1887].)

	Name	Description
	ContainsName [▶ 1900]	Determines whether the specified instance name contains name. (Inherited from InstanceCollection.T. [▶ 1887].)
	CopyTo [▶ 1901]	Copies this InstanceCollection.T. [▶ 1887] to the specified array. (Inherited from InstanceCollection.T. [▶ 1887].)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1902]	Gets the enumerator. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1902]	Gets the IInstance [▶ 1556]by instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetInstanceByName [▶ 1903]	Gets the name of the instance by. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1904]	Determines the index of the specified IInstance [▶ 1556]. (Inherited from InstanceCollection.T. [▶ 1887].)
	Insert [▶ 1905]	Inserts the specified IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1905]	Removes the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	RemoveAt [▶ 1906]	Removes the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1907]	Tries to get the IInstance [▶ 1556]. of the specified path. (Inherited from InstanceCollection.T. [▶ 1887].)
	TryGetInstanceByName [▶ 1908]	Tries to get Instnaces by name. (Inherited from InstanceCollection.T. [▶ 1887].)
	TryGetInstances [▶ 1969]	Try to get instances with predicate function

Fields

	Name	Description
	_list [▶ 1909]	The _list (Inherited from InstanceCollection.T. [▶ 1887].)
	_pathDict [▶ 1909]	The _path dictionary (Inherited from InstanceCollection.T. [▶ 1887].)
	mode [▶ 1910]	The mode this InstanceCollection.T. [▶ 1887] is working in. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.14.1 SymbolCollection.T. Properties

The SymbolCollection.T. [▶ 1964] generic type exposes the following members.

Properties

	Name	Description
	Count [▶ 1891]	Gets the collection count. (Inherited from InstanceCollection.T. [▶ 1887].)
	IsReadOnly [▶ 1892]	Gets a value indicating whether this instance is read only. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.Int32. [▶ 1893]	Gets or sets the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	Item.String. [▶ 1894]	Gets the IInstance [▶ 1556] with the specified instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	Mode [▶ 1894]	Gets the InstanceCollectionMode [▶ 1910]. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

SymbolCollection.T. Class [▶ 1964]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.14.2 SymbolCollection.T. Methods

The SymbolCollection.T. [▶ 1964] generic type exposes the following members.

Methods

	Name	Description
	Add [▶ 1896]	Adds the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	AddRange [▶ 1897]	Adds the specified items to this collection. (Inherited from InstanceCollection.T. [▶ 1887].)
	AsReadOnly [▶ 1968]	Returns a Read only version of this collection (shallow copy).
	Clear [▶ 1898]	Clears this instance. (Inherited from InstanceCollection.T. [▶ 1887].)
	Clone [▶ 1969]	Clones this instance.
	Contains(String) [▶ 1899]	Determines whether this collection contains an IInstance [▶ 1556] with the specified InstanceName / InstancePath (Inherited from InstanceCollection.T. [▶ 1887].)
	Contains(T) [▶ 1899]	Determines whether this collection contains the specified IInstance [▶ 1556] (Inherited from InstanceCollection.T. [▶ 1887].)
	ContainsName [▶ 1900]	Determines whether the specified instance name contains name. (Inherited from InstanceCollection.T. [▶ 1887].)
	CopyTo [▶ 1901]	Copies this InstanceCollection.T. [▶ 1887] to the specified array. (Inherited from InstanceCollection.T. [▶ 1887].)
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)

	Name	Description
	GetEnumerator [▶ 1902]	Gets the enumerator. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetInstance [▶ 1902]	Gets the IInstance [▶ 1556]by instance path. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetInstanceByName [▶ 1903]	Gets the name of the instance by. (Inherited from InstanceCollection.T. [▶ 1887].)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	IndexOf [▶ 1904]	Determines the index of the specified IInstance [▶ 1556]. (Inherited from InstanceCollection.T. [▶ 1887].)
	Insert [▶ 1905]	Inserts the specified IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	Remove [▶ 1905]	Removes the specified item. (Inherited from InstanceCollection.T. [▶ 1887].)
	RemoveAt [▶ 1906]	Removes the IInstance [▶ 1556] at the specified index. (Inherited from InstanceCollection.T. [▶ 1887].)
	ToString	Returns a string that represents the current object. (Inherited from Object.)
	TryGetInstance [▶ 1907]	Tries to get the IInstance [▶ 1556]. of the specified path. (Inherited from InstanceCollection.T. [▶ 1887].)
	TryGetInstanceByName [▶ 1908]	Tries to get Instnaces by name. (Inherited from InstanceCollection.T. [▶ 1887].)
	TryGetInstances [▶ 1969]	Try to get instances with predicate function

Reference

SymbolCollection.T. Class [▶ 1964]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.14.2.1 SymbolCollection.T..AsReadOnly Method

Returns a Read only version of this collection (shallow copy).

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public ReadOnlySymbolCollection<T> AsReadOnly()
```

VB

```
Public Function AsReadOnly As ReadOnlySymbolCollection(Of T)
```

Return Value

Type: ReadOnlySymbolCollection [▶ 1960].T [▶ 1964].
ReadOnlySymbolCollection<T>.

Reference

SymbolCollection.T. Class [▶ 1964]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.14.2.2 SymbolCollection.T..Clone Method

Clones this instance.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolCollection<T> Clone()
```

VB

```
Public Function Clone As SymbolCollection(Of T)
```

Return Value

Type: SymbolCollection [▶ 1964].T [▶ 1964].
SymbolCollection<T>.

Reference

SymbolCollection.T. Class [▶ 1964]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.14.2.3 SymbolCollection.T..TryGetInstances Method

Try to get instances with predicate function

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool TryGetInstances(
    Func<T, bool> predicate,
    bool recurse,
    out IList<T> instances
)
```

VB

```
Public Function TryGetInstances (
    predicate As Func(Of T, Boolean),
    recurse As Boolean,
    <OutAttribute> ByRef instances As IList(Of T)
) As Boolean
```

Parameters

predicate	Type: System.Func.T [▶ 1964], Boolean. The predicate function
recurse	Type: System.Boolean if set to true the symbol hierarchy will be searched recursively.
instances	Type: System.Collections.Generic.IList.T [▶ 1964].. The instances.

Return Value

Type: Boolean
true if XXXX, false otherwise.

Reference

SymbolCollection.T. Class [▶ 1964]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.14.3 SymbolCollection.T. Fields

The SymbolCollection.T. [▶ 1964] generic type exposes the following members.

Fields

	Name	Description
	_list [▶ 1909]	The _list (Inherited from InstanceCollection.T. [▶ 1887].)
	_pathDict [▶ 1909]	The _path dictionary (Inherited from InstanceCollection.T. [▶ 1887].)
	mode [▶ 1910]	The mode this InstanceCollection.T. [▶ 1887] is working in. (Inherited from InstanceCollection.T. [▶ 1887].)

Reference

SymbolCollection.T. Class [▶ 1964]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.15 SymbolIterationMask Enumeration

Mask Flagset to specify filters for SymbolIterator.T. [▶ 1971].

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[FlagsAttribute]
public enum SymbolIterationMask
```

VB

```
<FlagsAttribute>
Public Enumeration SymbolIterationMask
```

Members

	Member name	Value	Description
	None	0	Uninitialized / None
	Structures	1	Iterates over Subelements of Structs
	Arrays	2	Iterates over Elements of Arrays
	Unions	4	Iterates over Subelements of Unions
	Pointer	8	Iterates over Pointer SubElements
	References	16	Iterates over References
	All	31	Iterates over All Complex/Combined types

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.16 SymbolIterator.T. Class

Symbol iterator object

Inheritance Hierarchy

System.Object
  TwinCAT.TypeSystem.Generic.SymbolIterator.T.
Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public class SymbolIterator<T> : IEnumerable<T>,
    IEnumerable
where T : ISymbol
```

VB

```
Public Class SymbolIterator(Of T As ISymbol)
    Implements IEnumerable(Of T), IEnumerable
```

Type Parameters

T — Concrete ISymbol [▶ 1634] type.

The SymbolIterator.T. type exposes the following members.

Constructors

	Name	Description
	SymbolIterator.T. (IInstanceCollection.T.) [▶ 1973]	Initializes a new instance of the SymbolIterator.T. class.
	SymbolIterator.T. (IEnumerable.T., Boolean) [▶ 1974]	Initializes a new instance of the SymbolIterator.T. class.

	Name	Description
	SymbolIterator.T. (IInstanceCollection.T., Func.T, Boolean.) [▶ 1974]	Initializes a new instance of the SymbolIterator.T. class.
	SymbolIterator.T. (IEnumerable.T., Boolean, Func.T, Boolean.) [▶ 1975]	Initializes a new instance of the SymbolIterator.T. class.

Properties

	Name	Description
	Mask [▶ 1976]	Gets or sets the SymbolIterationMask [▶ 1970]
	SymbolRecursionDetection [▶ 1976]	Gets or sets a value indicating whether the iterator checks for Symbol recursions (true by default).

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1977]	Gets the enumerator that enumerates through a collection
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Remarks

Iterates over all root symbols and its sub symbols.

Reference

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

System.Collections.Generic.IEnumerable.T.

5.9.16.1 SymbolIterator.T. Constructor

Overload List

	Name	Description
	SymbolIterator.T. (IInstanceCollection. T.) [▶ 1973]	Initializes a new instance of the SymbolIterator.T. [▶ 1971] class.
	SymbolIterator.T. (IEnumerable.T., Boolean) [▶ 1974]	Initializes a new instance of the SymbolIterator.T. [▶ 1971] class.
	SymbolIterator.T. (IInstanceCollection. T., Func.T, Boolean.) [▶ 1974]	Initializes a new instance of the SymbolIterator.T. [▶ 1971] class.
	SymbolIterator.T. (IEnumerable.T., Boolean, Func.T, Boolean.) [▶ 1975]	Initializes a new instance of the SymbolIterator.T. [▶ 1971] class.

Reference

SymbolIterator.T. Class [▶ 1971]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.16.1.1 SymbolIterator.T. Constructor (IInstanceCollection.T.)

Initializes a new instance of the SymbolIterator.T. [▶ 1971] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolIterator(
    IInstanceCollection<T> coll
)
```

VB

```
Public Sub New (
    coll As IInstanceCollection(Of T)
)
```

Parameters

coll	Type: TwinCAT.TypeSystem.Generic.IInstanceCollection [▶ 1877].T [▶ 1971]. The root symbols.

Reference

SymbolIterator.T. Class [▶ 1971]

SymbolIterator.T. Overload [▶ 1973]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.16.1.2 SymbolIterator.T. Constructor (IEnumerable.T., Boolean)

Initializes a new instance of the SymbolIterator.T. [▶ 1971] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolIterator(
    IEnumerable<T> coll,
    bool recurse
)
```

VB

```
Public Sub New (
    coll As IEnumerable(Of T),
    recurse As Boolean
)
```

Parameters

coll	Type: System.Collections.Generic.IEnumerable.T [▶ 1971]. The root collection
recurse	Type: System.Boolean if set to true the iterator will iterate recursively.

Reference

SymbolIterator.T. Class [▶ 1971]

SymbolIterator.T. Overload [▶ 1973]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.16.1.3 SymbolIterator.T. Constructor (IInstanceCollection.T., Func.T, Boolean.)

Initializes a new instance of the SymbolIterator.T. [▶ 1971] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolIterator(
    IInstanceCollection<T> coll,
    Func<T, bool> predicate
)
```

VB

```
Public Sub New (
    coll As IInstanceCollection(Of T),
    predicate As Func(Of T, Boolean)
)
```

Parameters

coll	Type: TwinCAT.TypeSystem.Generic.IInstanceCollection [▶ 1877].T [▶ 1971]. The root symbols.
predicate	Type: System.Func.T [▶ 1971], Boolean. The predicate.

Exceptions

Exception	Condition
NotSupportedException	

Reference

SymbolIterator.T. Class [▶ 1971]

SymbolIterator.T. Overload [▶ 1973]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.16.1.4 SymbolIterator.T. Constructor (IEnumerable.T., Boolean, Func.T, Boolean.)

Initializes a new instance of the SymbolIterator.T. [▶ 1971] class.

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolIterator(
    IEnumerable<T> coll,
    bool recurse,
    Func<T, bool> predicate
)
```

VB

```
Public Sub New (
    coll As IEnumerable(Of T),
    recurse As Boolean,
    predicate As Func(Of T, Boolean)
)
```

Parameters

coll	Type: System.Collections.Generic.IEnumerable.T [▶ 1971]. Input collection (root objects).
recurse	Type: System.Boolean if set to true the iterator will iterate recursively.
predicate	Type: System.Func.T [▶ 1971], Boolean. The predicate.

Reference

SymbolIterator.T. Class [▶ 1971]

SymbolIterator.T. Overload [▶ 1973]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.16.2 SymbolIterator.T. Properties

The SymbolIterator.T. [▶ 1971] generic type exposes the following members.

Properties

	Name	Description
	Mask [▶ 1976]	Gets or sets the SymbolIterationMask [▶ 1970]
	SymbolRecursionDetection [▶ 1976]	Gets or sets a value indicating whether the iterator checks for Symbol recursions (true by default).

Reference

SymbolIterator.T. Class [▶ 1971]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.16.2.1 SymbolIterator.T..Mask Property

Gets or sets the SymbolIterationMask [▶ 1970]

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public SymbolIterationMask Mask { get; set; }
```

VB

```
Public Property Mask As SymbolIterationMask
    Get
    Set
```

Property Value

Type: SymbolIterationMask [▶ 1970]
The mask.

Remarks

This property can be used for prefiltering the iterator without using a predicate function.

Reference

SymbolIterator.T. Class [▶ 1971]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.16.2.2 SymbolIterator.T..SymbolRecursionDetection Property

Gets or sets a value indicating whether the iterator checks for Symbol recursions (true by default).

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public bool SymbolRecursionDetection { get; set; }
```

VB

```
Public Property SymbolRecursionDetection As Boolean
    Get
    Set
```

Property Value

Type: Boolean
true if recursion checking, false switched off check.

Reference

SymbolIterator.T. Class [▶ 1971]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.16.3 SymbolIterator.T. Methods

The SymbolIterator.T. [▶ 1971] generic type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object.)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
	GetEnumerator [▶ 1977]	Gets the enumerator that enumerates through a collection
	GetHashCode	Serves as the default hash function. (Inherited from Object.)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	MemberwiseClone	Creates a shallow copy of the current Object. (Inherited from Object.)
	ToString	Returns a string that represents the current object. (Inherited from Object.)

Reference

SymbolIterator.T. Class [▶ 1971]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.9.16.3.1 SymbolIterator.T..GetEnumerator Method

Gets the enumerator that enumerates through a collection

Namespace: TwinCAT.TypeSystem.Generic [▶ 1854]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
public IEnumerator<T> GetEnumerator()
```

VB

```
Public Function GetEnumerator As IEnumerator(Of T)
```

Return Value

Type: IEnumerator.T [▶ 1971].
A IEnumerator.T. that can be used to iterate through the collection.

Implements

IEnumerable.T..GetEnumerator.

Reference

SymbolIterator.T. Class [▶ 1971]

TwinCAT.TypeSystem.Generic Namespace [▶ 1854]

5.10 TwinCAT.ValueAccess Namespace

Namespace for the common (non ADS dependant) value access.

Enumerations

	Enumeration	Description
	SymbolNotificationType [▶ 1978]	Specifies the Notification type of ADS Notifications
	ValueCreationMode [▶ 1979]	Creation mode for Values

5.10.1 SymbolNotificationType Enumeration

Specifies the Notification type of ADS Notifications

Namespace: TwinCAT.ValueAccess [▶ 1978]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[FlagsAttribute]
public enum SymbolNotificationType
```

VB

```
<FlagsAttribute>
Public Enumeration SymbolNotificationType
```

Members

	Member name	Value	Description
	None	0	None / Uninitialized
	Value	1	ValueChanged notifications
	RawValue	2	RawValueChanged notifications
	Both	3	ValueChanged + RawValueChanged notifications

Reference

TwinCAT.ValueAccess Namespace [▶ 1978]

5.10.2 ValueCreationMode Enumeration

Creation mode for Values

Namespace: TwinCAT.ValueAccess [▶ 1978]
Assembly: TwinCAT.Ads (in TwinCAT.Ads.dll) Version: 4.3.0.0

Syntax

C#

```
[FlagsAttribute]
public enum ValueCreationMode
```

VB

```
<FlagsAttribute>
Public Enumeration ValueCreationMode
```

Members

	Member name	Value	Description
	None	0	None / Uninitialized Mask
	Primitives	1	Convert to .NET Primitives, wherever possible.
	Enums	2	Use IEnumValue on EnumTypes instead of .NET Primitives
	FullDynamics	4	Wraps all Primitives also in IValue Objects
	PlcOpenTypes	8	Use PlcOpen Times (TIME, LTIME, DT, TOD, DATETIME) instead of .NET Primitives DateTime and TimeSpan
	Default	1	Default settings for the value creation mode (Translate to Primitives)

Remarks

This setting is used by the ValueFactory/ to create Read values. In default primitive mode all values will be transferred to Primitive .NET Symbols if possible. E.g PlcOpen.TIME --> TimeSpan, IEnumValue --> .NET Primitives. They won't be wrapped into

Reference

TwinCAT.ValueAccess Namespace [▶ 1978]

6 Samples

6.1 Basic Samples

The topics in this section describe the various changes made to the 'TwinCAT.Ads .NET API' over the life of the project.

HowTo

How to use the .NET TwinCAT API

Sample	Description
Reading/writing (simple) values from/to ADS Servers.	Read/Write values [▶ 1981]
Reading/writing string values.	Read/Write string values [▶ 1983]
Reading/writing PlcOpen values within the TwinCAT Ads Communication Library	Read/Write PlcOpen types (DATE, TIME ...) [▶ 1984]
Event driven reading (ADS Notifications)	Event driven reading [▶ 1985]
Read/Write data from/to ADS Servers using reactive extensions	Reactive Read/Write [▶ 1986]

Other Resources

TwinCAT ADS .NET API Documentation [▶ 16]

6.1.1 Read/Write primitive values

Reading writing Values from ADS Devices is the most essential part of the communication API. There are several options for communication with your application.

- Accessing by IndexGroup / IndexOffset
- Symbolic access by instance path and optionally use handles for the symbol
- Holding the overall Symbolic information in the SymbolLoader and use easy access via symbol objects.
- Usage the symbolic interface ITcAdsSymbol when complete SymbolLoading by SymbolLoaders is not appropriate.
- Reading / Writing values as .NET managed Types (primitive types or compound primitive types called ANY_TYPES), or complex dynamic types typesafe generated at runtime.

The following section shows the different scenarios as code snippets.

HowTo Read/Write Values

Read/Write AnyType by IndexGroup/IndexOffset

```
using (TcAdsClient client = new TcAdsClient())
{
    UInt32 valueToRead = 0;
    UInt32 valueToWrite = 42;

    client.Connect(AmsNetId.Local, 851);
```

```
    adsClient.WriteAny(0x4020, 0x0, valueToWrite);
    valueToRead = (uint)adsClient.ReadAny(0x4020, 0x0, typeof(UInt32));
}
```

Read/Write AnyType by variable handle

```
using (TcAdsClient client = new TcAdsClient())
{
    int varHandle = 0;
    client.Connect(AmsNetId.Local, 851);
    try
    {
    UInt32 valueToRead = 0;
    UInt32 valueToWrite = 42;

    varHandle = client.CreateVariableHandle("MAIN.nCounter");
    adsClient.WriteAny(varHandle, valueToWrite);
    valueToRead = (uint)adsClient.ReadAny(varHandle, typeof(UInt32));
    }
    finally
    {
    // Unregister VarHandle after Use
    client.DeleteVariableHandle(varHandle);
    }
}
```

Read/Write AnyType by instance/symbol path

```
using (TcAdsClient client = new TcAdsClient())
{
    UInt32 valueToRead = 0;
    UInt32 valueToWrite = 42;

    client.Connect(AmsNetId.Local, 851);
    adsClient.WriteSymbol("MAIN.nCounter", valueToWrite, false);
    valueToRead = (uint)adsClient.ReadSymbol("MAIN.nCounter", typeof(UInt32),false);
}
```

Read/Write AnyType by IAdsSymbol

```
using (TcAdsClient client = new TcAdsClient())
{
    UInt32 valueToRead = 0;
    UInt32 valueToWrite = 42;

    client.Connect(AmsNetId.Local, 851);

    ITcAdsSymbol symbol = adsClient.ReadSymbolInfo("MAIN.nCounter");
    adsClient.WriteSymbol(symbol, valueToWrite);
    valueToRead = (uint)adsClient.ReadSymbol(symbol);
}
```

Read/Write AnyType by SymbolBrowser

```
using (TcAdsClient client = new TcAdsClient())
{
    UInt32 valueToRead = 0;
    UInt32 valueToWrite = 42;

    client.Connect(AmsNetId.Local, 851);

    // Load all Symbols + DataTypes
    ISymbolLoader loader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    Symbol symbol = (Symbol)loader.Symbols["MAIN.nCounter"];

    // Works for ALL Primitive 'ANY TYPES' Symbols
    symbol.WriteValue(valueToWrite);
    valueToRead = (UInt32)symbol.ReadValue();
}
```

Read/Write dynamic types by SymbolBrowser

```
using (TcAdsClient client = new TcAdsClient())
{
    UInt32 valueToRead = 0;
    UInt32 valueToWrite = 42;
```

```
    client.Connect(AmsNetId.Local, 851);

    // Load all Symbols + DataTypes
    // Primitive Parts will be automatically resolved to .NET Primitive types.
    IDynamicSymbolLoader loader = (IDynamicSymbolLoader)SymbolLoaderFactory.Create(client, SymbolLoa
derSettings.DefaultDynamic);

    dynamic symbols = loader.SymbolsDynamic;
    dynamic main = symbols.Main;

    // Use typed object to use InfoTips
    DynamicSymbol nCounter = main.nCounter;

    // or to be fullDynamic
    //dynamic nCounter = main.nCounter;

    // Works for ALL sorts of types (not restricted to ANY_TYPE basing primitive types)
    nCounter.WriteValue(valueToWrite);
    valueToRead = (uint)nCounter.ReadValue();
}
```

6.1.2 Read/Write string types

ADS Server usually support strings in 2 flavors. The Default (ANSI) and the Unicode encoding (STRING vs. WSTRING) The ANSI encoding reserves 1 byte per character. Unicode reserves 2.

The strings are of fixed size and therefore the length of the the reserved space within the process image is important.

HowTo Read/Write string values

Reading writing ANSI Streams:

Read/Write ANSI Strings

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local port 851 (PLC)

    int handle = client.CreateVariableHandle("MAIN.string"); // Symbol "string" in MAIN defined as s
tring

    try
    {
    // Read ANSI String string[80]
    int byteSize = 81; // Size of 80 ANSI chars + /0 (STRING[80])
    AdsStream readStream = new AdsStream(byteSize); // Size of 80 ANSI chars + /0 (STRING[80])
    AdsBinaryReader reader = new AdsBinaryReader(readStream);
    client.Read(handle, readStream, 0, byteSize); // Read 81 bytes
    string value = reader.ReadPlcString(byteSize,Encoding.Default);

    // Write ANSI String string[80]
    AdsStream writeStream = new AdsStream(byteSize);
    AdsBinaryWriter writer = new AdsBinaryWriter(writeStream);
    value = "Changed";
    writer.WritePlcString(value, 80,Encoding.Default); // Max 80 characters!
    client.Write(handle, writeStream, 0, byteSize);
    }
    finally
    {
    client.DeleteVariableHandle(handle);
    }
}
```

Reading writing UNICODE Streams:

Read/Write Unicode Strings

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local port 851 (PLC)

    int handle = client.CreateVariableHandle("MAIN.wstring"); // Symbol "wstring" defined in MAIN as
 WSTRING
```

```
    try
    {
    // Read UNICODE String wstring[80]
    int byteSize = 2 * 81; // Size of 80 UNICODE chars + /0 (WSTRING[80])
    AdsStream readStream = new AdsStream(byteSize);
    AdsBinaryReader reader = new AdsBinaryReader(readStream);
    client.Read(handle, readStream, 0, byteSize); // Read 2*81 bytes
    string value = reader.ReadPlcString(byteSize,Encoding.Unicode);

    // Write ANSI String string[80]
    AdsStream writeStream = new AdsStream(byteSize);
    AdsBinaryWriter writer = new AdsBinaryWriter(writeStream);
    value = "Changed";
    writer.WritePlcString(value, 80,Encoding.Unicode);
    client.Write(handle, writeStream, 0, byteSize);
    }
    finally
    {
    client.DeleteVariableHandle(handle);
    }
}
```

Reading writing strings with ReadAny/WriteAny group of methods:

Read/Write Unicode Strings

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local port 851 (PLC)

    int stringHandle = client.CreateVariableHandle("MAIN.string"); // Symbol "string" defined in MAI
N as STRING
    int wStringHandle = client.CreateVariableHandle("MAIN.wstring"); // Symbol "string" defined in M
AIN as WSTRING

    try
    {
    string str = client.ReadAnyString(stringHandle, 80, Encoding.Default);
    string wStr = client.ReadAnyString(wStringHandle, 80, Encoding.Unicode);

    string changedValue = "Changed";

    // Attention, take care that the memory of the string in the process image is not exceeded!
    client.WriteAnyString(stringHandle, changedValue, 80, Encoding.Default);
    client.WriteAnyString(wStringHandle, changedValue, 80, Encoding.Unicode);
    }
    finally
    {
    client.DeleteVariableHandle(stringHandle);
    client.DeleteVariableHandle(wStringHandle);
    }
}
```

6.1.3 Read/Write PlcOpen types (DATE, TIME ...)

The following PLCOpen types have specific representations within the TwinCAT.Ads Communication Library: DT

- DT
- DATE [▶ 1121]
- TIME [▶ 1158]
- LTIME [▶ 1139]
- TOD [▶ 1176]

The following section shows the different scenarios as code snippets.

HowTo Read/Write PlcOpen values

Reading writing by streams:

Read/Write PlcOpen types (streamed)

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local plc

    int handle1 = 0;
    int handle2 = 0;

    try
    {
    handle1 = client.CreateVariableHandle("MAIN.time"); // TIME
    handle2 = client.CreateVariableHandle("MAIN.date"); // DATE

    AdsStream readStream = new AdsStream(LTIME.MarshalSize); // Largest (8 Bytes)
    AdsBinaryReader reader = new AdsBinaryReader(readStream);

    client.Read(handle1, readStream, 0, TIME.MarshalSize);
    TimeSpan time = reader.ReadPlcTIME();

    client.Read(handle2, readStream, 0, DATE.MarshalSize);
    DateTime dateTime = reader.ReadPlcDATE();
    }
    finally
    {
    client.DeleteVariableHandle(handle1);
    client.DeleteVariableHandle(handle2);
    }
}
```

Reading writing by ANY type concept:

Read/Write PlcOpen types (ANY)

```
using (TcAdsClient client = new TcAdsClient())
{
    client.Connect(851); // Connect to local plc

    int handle1 = 0;
    int handle2 = 0;
    int handle3 = 0;

    try
    {
    handle1 = client.CreateVariableHandle("MAIN.time"); // TIME
    handle2 = client.CreateVariableHandle("MAIN.date"); // DATE
    handle3 = client.CreateVariableHandle("MAIN.ltime"); // LTIME

    TIME time = (TIME)client.ReadAny(handle1, typeof(TIME)); // TIME
    TimeSpan timeSpan = time.Time;

    DATE date = (DATE)client.ReadAny(handle2, typeof(DATE)); // DATE
    DateTime dateTime = date.Date;

    LTIME ltime = (LTIME)client.ReadAny(handle3, typeof(LTIME)); // LTIME
    TimeSpan lTimeSpan = ltime.Time;
    }
    finally
    {
    client.DeleteVariableHandle(handle1);
    client.DeleteVariableHandle(handle2);
    client.DeleteVariableHandle(handle3);
    }
}
```

6.1.4 Event driven read with ADS Notifications

Use of ADS Notifications

Trigger on changed values by ADS Notifications

```
AdsStream readStream = new AdsStream(sizeof(UInt32));

private void ReceiveNotifications()
{
```

```
    using (TcAdsClient client = new TcAdsClient())
    {
    // Add the Notification event handler
    client.AdsNotification += Client_AdsNotification;

    // Connect to target
    client.Connect("1.2.3.4.5.6", 851);
    int notificationHandle = 0;

    try
    {
        // Notification to a DINT Type (UINT32)
        // Check for change every 200 ms
        notificationHandle = client.AddDeviceNotification("MAIN.nCounter", readStream, AdsTransMode.
OnChange, 200, 0, null);
        Thread.Sleep(5000); // Sleep the main thread to get some (asynchronous Notifications)
    }
    finally
    {
        // Unregister the Event / Handle
        client.DeleteDeviceNotification(notificationHandle);
        client.AdsNotification -= Client_AdsNotification;
    }
    }
}

private void Client_AdsNotification(object sender, AdsNotificationEventArgs e)
{
    int offset = (int)e.DataStream.Position;
    int length = (int)e.DataStream.Length;

    e.DataStream.Position = offset;
    AdsBinaryReader reader = new AdsBinaryReader(e.DataStream);

    // Read the Unmarshalled data
    //byte[] data = reader.ReadBytes(length);

    // Or here we know about UDINT type --> can be marshalled as UINT32
    uint nCounter = reader.ReadUInt32();
}
```

6.1.5 Reactive Read/Write with Reactive Extensions

Observation of Notifications

Notifications (address specified by InstancePath) will be received cyclically as defined in Default [▶ 576] and put into the Observer pipeline for further processing. This example takes 20 Notification samples before returning.

Observe for Notifications

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Reactive Notification Handler
    var valueObserver = Observer.Create<ushort>(val =>
    {
    Console.WriteLine(string.Format("Value: {0}", val.ToString()));
    }
    );

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    IDisposable subscription = client.WhenNotification<ushort>("TwinCAT_SystemInfoVarList._TaskInfo.
CycleCount", NotificationSettings.Default).Take(20).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Observation of Symbolic Notifications

This example determines a symbol via Symbolloader and samples its values by Notifications with customized NotificationSettings [▶ 573]. Again 20 samples are taken before the Observation finishes.

Observe for Symbol Notifications

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o[1].CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    cycleCount.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 500, 5000); //
 optional: Change NotificationSettings on Symbol

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    IDisposable subscription = cycleCount.WhenValueChanged().Take(20).Subscribe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Observation of dynamic Symbol Notifications

Here, the symbol is determined via Symbolloader again, but now the Notifications will be processed as 'dynamic' values.

Observer for dynamic Symbol Notifications

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = (IDynamicSymbolLoader)SymbolLoaderFactory.Create(client, SymbolLoaderSettings
.DefaultDynamic);
    dynamic symbols = symbolLoader.SymbolsDynamic;
    dynamic cycleCount = symbols.TwinCAT_SystemInfoVarList._TaskInfo[1].CycleCount;

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    // Value objects can be dynamically (on the fly) created objects here (e.g. structs)
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    cycleCount.NotificationSettings = new NotificationSettings(AdsTransMode.OnChange, 500, 5000); //
 optional: Change NotificationSettings on Symbol

    // Turning ADS Notifications into sequences of Value Objects (Taking 20 Values)
    // and subscribe to them.
    // We have to give the 'hint' about IValueSymbol here, that the CLR finds the Extension Method '
WhenValueChanged' during runtime.
    IDisposable subscription = ((IValueSymbol)cycleCount).WhenValueChanged().Take(20).Subscribe(valu
eObserver);
```

```
    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Polling observer

A polling observer doesn't use ADS notifications, but instead the value read is triggered by a time interval (polling) or a customized trigger function (on request).

Polling observer

```
// To Test the Observer run a project on the local PLC System (Port 851)
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol cycleCount = (IValueSymbol)symbolLoader.Symbols["TwinCAT_SystemInfoVarList._TaskInf
o.CycleCount"];

    // Reactive Notification Handler
    var valueObserver = Observer.Create<object>(val =>
    {
    Console.WriteLine(string.Format("Instance: {0}, Value: {1}", cycleCount.InstancePath, val.ToStri
ng()));
    }
    );

    // Take 20 Values in an Interval of 500ms
    IDisposable subscription = cycleCount.PollValues(TimeSpan.FromMilliseconds(500)).Take(20).Subscr
ibe(valueObserver);

    Console.ReadKey(); // Wait for Key press
    subscription.Dispose(); // Dispose the Subscription
}
```

Writing values with observable subject

In this example, a symbolic value is written in a static time interval (1 second). The writing stops after 10 values. The WriteValues [▶ 888] extension method can be used to seamlessly bind value writing into a reactive application.

Writing values with observable subject

```
using (TcAdsClient client = new TcAdsClient())
{
    // Connect to target
    client.Connect(new AmsAddress(AmsNetId.Local, 851));

    // Create Symbol information (Symbol 'i : INT' in PLC Global Variables list.
    var symbolLoader = SymbolLoaderFactory.Create(client, SymbolLoaderSettings.Default);
    IValueSymbol gvlIntSymbol = (IValueSymbol)symbolLoader.Symbols["GVL.i"];

    // Produces object (short) Values 0,1,2,3 ... in seconds period
    IObservable<object> timerObservable = Observable.Interval(TimeSpan.FromSeconds(1.0)).Select(i =>
 (object)(short)i);

    // Take 10 Values (0..9) and write them to GVL.i
    IDisposable dispose = gvlIntSymbol.WriteValues(timerObservable.Take(10));

    Console.ReadKey(); // Wait for Key press
    dispose.Dispose(); // Dispose the Subscription
}
```

6.2 Advanced Samples

Table 1: TwinCAT ADS .NET

Descripion
Linking into TwinCAT3 (C#) [▶ 1991]
Linking into Microsoft Visual Studio (C#) [▶ 1993]
Linking into Microsoft Visual Studio .NET (VB) [▶ 1996]
Linking into Borland Developer Studio 2006 (Delphi for .NET) [▶ 1998]
Linking into Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) [▶ 2001]

Description	Visual C#	Visual Basic (for .NET Framework)	Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	Delphi for .NET (Borland Developer Studio 2006)
Sample 1: Accessing an array in the PLC [▶ 2004]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490113291/.zip	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490114699/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490116107/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490117515/.exe
Sample 2: Transmitting a structure to the PLC [▶ 2011]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490118923/.zip	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490120331/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490121739/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490123147/.exe
Sample 3: Event driven reading [▶ 2019]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490124555/.zip	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490125963/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490127371/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490128779/.exe
Sample 4: Reading and writing of string variables [▶ 2029]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490130187/.zip	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490131595/.exe **TwinCAT.Ads.NET v1.0.0.10 and higher:** https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490133003/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490134411/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490135819/.exe
Sample 5: Reading and writing of DATE/TIME variables [▶ 2037]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490137227/.zip	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490138635/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490140043/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490141451/.exe

Description	Visual C#	Visual Basic (for .NET Framework)	Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	Delphi for .NET (Borland Developer Studio 2006)
Sample 6: Read PLC variable declaration [▶ 2045]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490142859/.zip	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490144267/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490145675/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490147083/.exe
Sample 7: Reading and writing of PLC variables of any type [▶ 2059]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490148491/.zip	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490149899/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490151307/.exe	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490152715/.exe
Sample 8: Detect state changes of TwinCAT router and PLC [▶ 2078]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490154123/.zip	-	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490155531/.exe	-
Sample 9: ADS-Sum Command: Reading or writing several variables [▶ 2082]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490156939/.zip	-	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490158347/.exe	-
Sample 10: Reading of SMB values from TwinCAT I/O driver [▶ 2095]	-	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490159755/.exe	-	-
Sample 11: Delete a handle of a PLC variable [▶ 2098]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490161163/.zip	-	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490162571/.exe	-
Sample 12: Read flag synchronously from the PLC [▶ 2100]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490163979/.zip	-	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490165387/.exe	-
Sample 13: Write flag synchronously into the PLC [▶ 2102]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490166795/.zip	-	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490168203/.exe	-
Sample 14: Start/stop PLC [▶ 2103]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490169611/.zip	-	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490171019/.exe	-

Description	Visual C#	Visual Basic (for .NET Framework)	Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	Delphi for .NET (Borland Developer Studio 2006)
Sample 15: Access by variable name [▶ 2105]	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490172427/.zip	-	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490173835/.exe	-

TwinCAT ADS Webservice

Description	Sample code
Sample 1: https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490175243/.htm	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490178187/.exe
Sample 1: Consumer in Delphi 8 for .NET to read and write PLC variables	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490179595/.exe

6.2.1 Linking the TwinCAT ADS .NET Component

6.2.1.1 Linking into TwinCAT3 (C#)

Creating new project

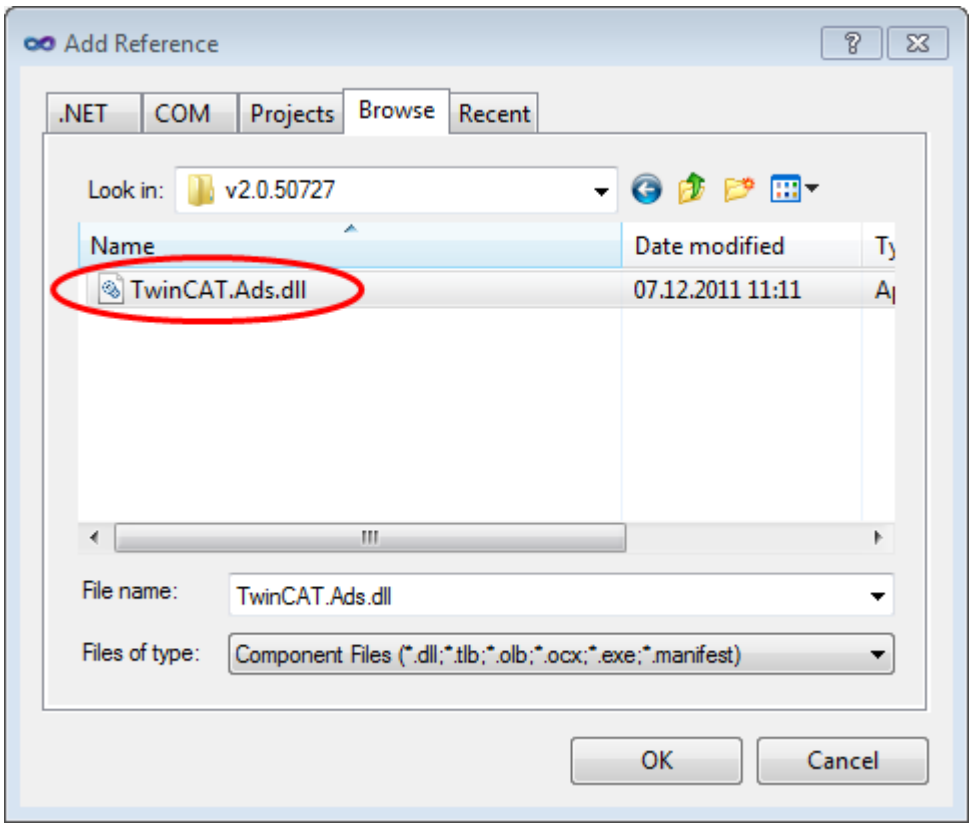
Start Microsoft Visual Studio and create new project (Windows Forms Application).

Adding reference

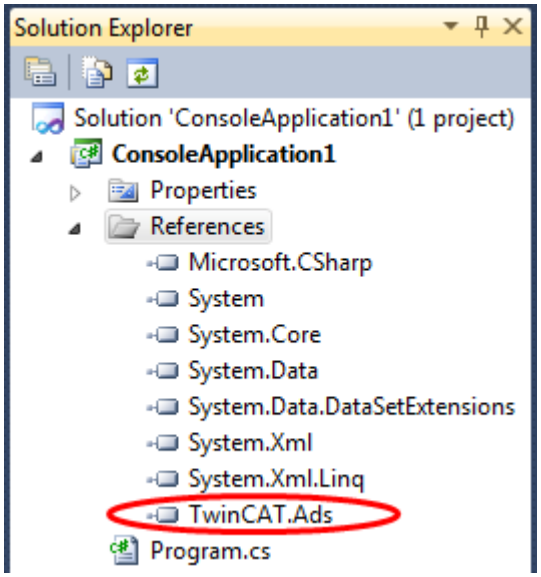
In order to select the TwinCAT.Ads class library you must choose the command *Add Reference...* under the *Project* menu. You will find the .Net Library per default in following TwinCAT folder:

C:\TwinCAT3\AdsApi\.NET\v2.0.50727

This opens the *Add Reference* dialog:



In this dialog you must press the *Browse* button and select the file TwinCAT.Ads.dll. In the Solution Explorer you can check if the component has been added to the list of references.



All accessible types (classes, structures ...) belong to the namespace TwinCAT.Ads. Therefore one has to insert the following line at the beginning of the source :

```
using System.IO, TwinCAT.Ads;
```

This enables access to the types defined in TwinCAT.Ads without including the name of the namespace. The class TcAdsClient is the core of the TwinCAT.Ads class library and enables the user to communicate with an ads device. To begin with an instance of the class must be created. Then a connection to the ADS device is established by means of the Connect method.

Establish a connection to an ADS device:

```
TcAdsClient tcAds = new TcAdsClient();
tcAds.Connect("127.0.0.1.1.1",851);
```

The data is transmitted with the help of the AdsStream class, that inherits form System.IO.MemoryStream. To write data to the stream one can utilize the System.IO.BinaryWriter class and the System.IO.BinaryReader class is necessary to read data from the stream.

Reading a PLC variable:

```
// creates a stream with a length of 4 byte
AdsStream ds = new AdsStream(4);
BinaryReader br = new BinaryReader(ds);

// reads a DINT from PLC
tcAds.Read(0x4020,0,ds);

ds.Position = 0;
textBox1.Text = br.ReadInt32().ToString();
```

Writing to a PLC variable:

```
// creates a stream with a length of 4 byte
AdsStream ds = new AdsStream(4);
BinaryWriter bw = new BinaryWriter(ds);

ds.Position = 0;
bw.Write((int) 100);

// writes a DINT to PLC
tcAds.Write(0x4020,0,ds);
```

Disconnect a connection to an ADS device:

```
tcAds.Dispose();
```

PLC Program:

```
PROGRAM MAIN
VAR
    test AT%MD0 : DINT;
END_VAR
```

```
test := test + 1;
```

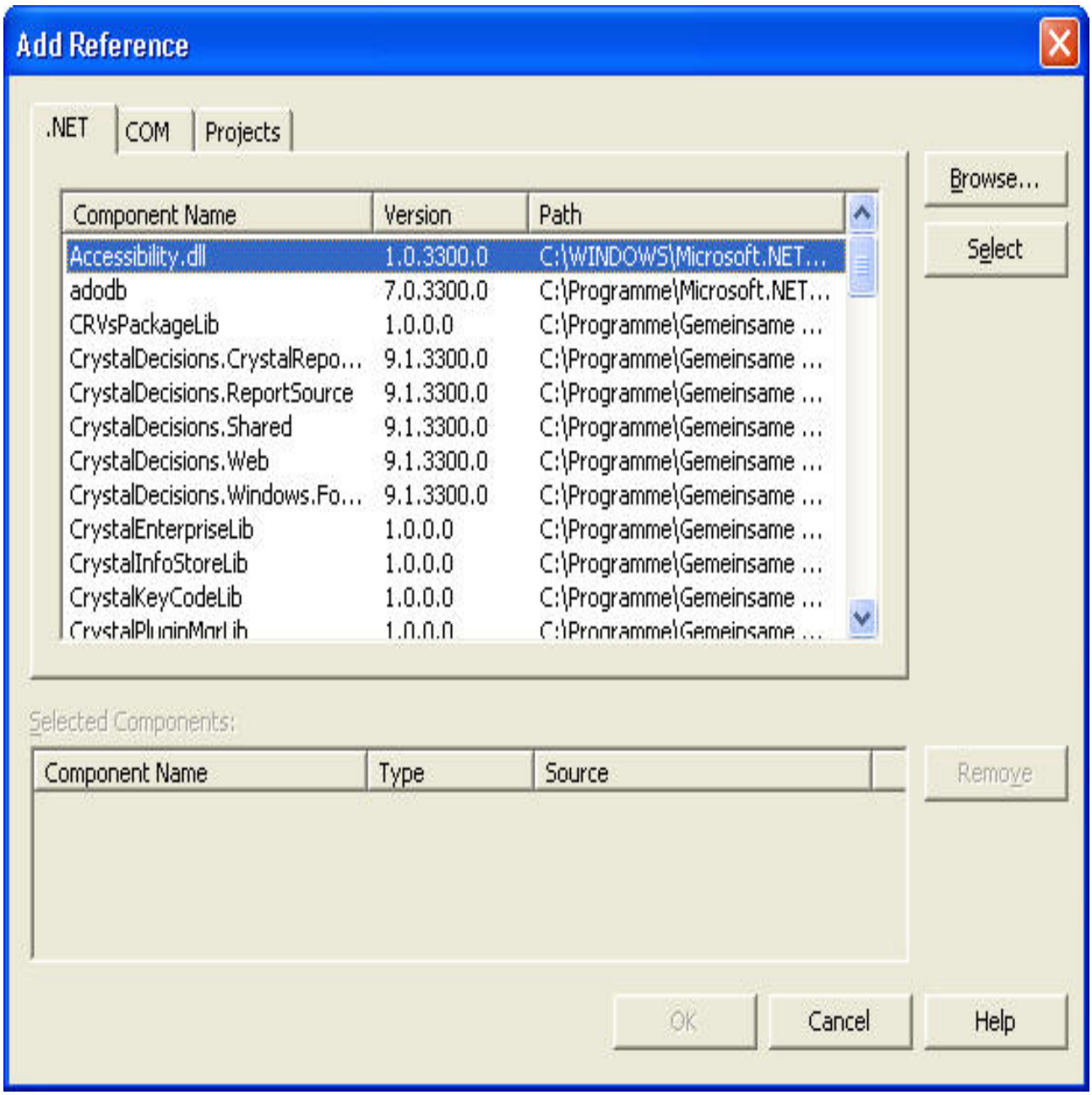
6.2.1.2 Linking into Microsoft Visual Studio .NET (C#)

Creating new project

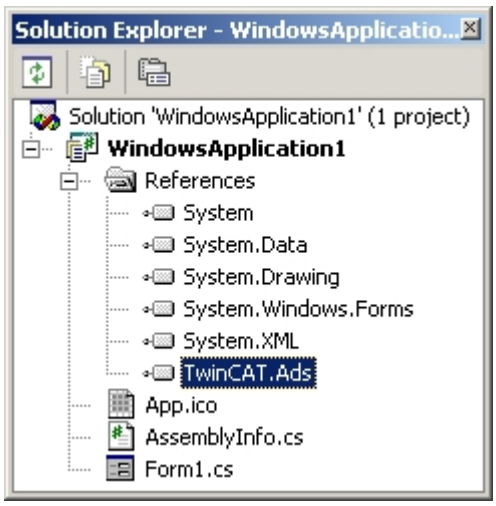
Start Microsoft Visual Studio and create new project (Windows Forms Application).

Adding reference

To select the TwinCAT.Ads class library you must choose the command *Add Reference...* under the *Project* menu. This opens the *Add Reference* dialog:



In this dialog you must press the *Browse* button and select the file TwinCAT.Ads.dll. In the Solution Explorer you can check if the component has been added to the list of references.



All accessible types (classes,structures ...) belong to the namespace TwinCAT.Ads. Therefore one has to insert the following line at the beginning of the source :

```
using System.IO, TwinCAT.Ads;
```

This enables access to the types defined in TwinCAT.Ads without including the name of the namespace. The class TcAdsClient is the core of the TwinCAT.Ads class library and enables the user to communicate with an ads device. To begin with an instance of the class must be created. Then a connection to the ADS device is established by means of the Connect method.

Establish a connection to an ADS device:

```
TcAdsClient tcAds = new TcAdsClient();
tcAds.Connect("172.16.3.217.1.1",801);
```

The data is transmitted with the help of the AdsStream class, that inherits form System.IO.MemoryStream. To write data to the stream one can utilize the System.IO.BinaryWriter class and the System.IO.BinaryReader class is necessary to read data from the stream.

Reading a PLC variable:

```
// creates a stream with a length of 4 byte
AdsStream ds = new AdsStream(4);
BinaryReader br = new BinaryReader(ds);

// reads a DINT from PLC
tcAds.Read(0x4020,0,ds);

ds.Position = 0;
textBox1.Text = br.ReadInt32().ToString();
```

Writing to a PLC variable:

```
// creates a stream with a length of 4 byte
AdsStream ds = new AdsStream(4);
BinaryWriter bw = new BinaryWriter(ds);

ds.Position = 0;
bw.Write((int) 100);

// writes a DINT to PLC
tcAds.Write(0x4020,0,ds);
```

Disconnect a connection to an ADS device:

```
tcAds.Dispose();
```

PLC Program:

```
PROGRAM MAIN
VAR
    test AT%MD0 : DINT;
END_VAR
```

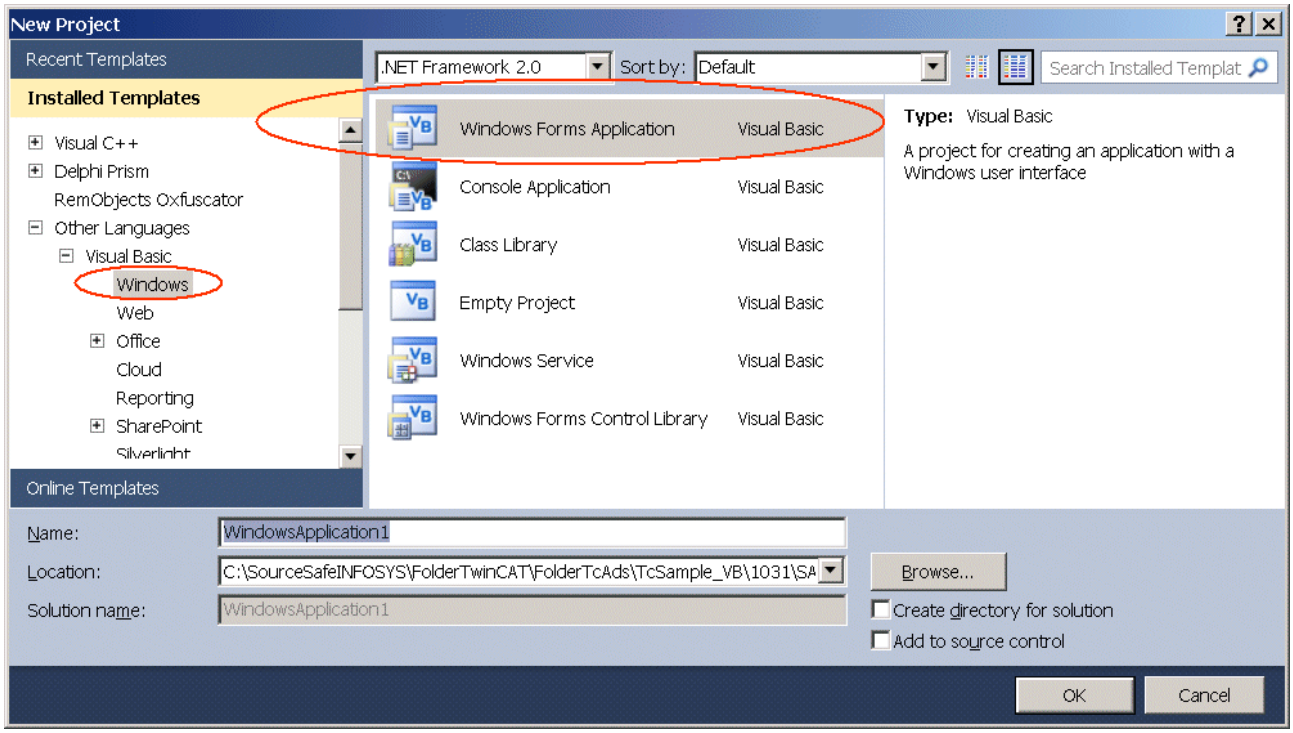
```
test := test + 1;
```

6.2.1.3 Integration in Microsoft Visual Studio .NET (VB)

Create new project

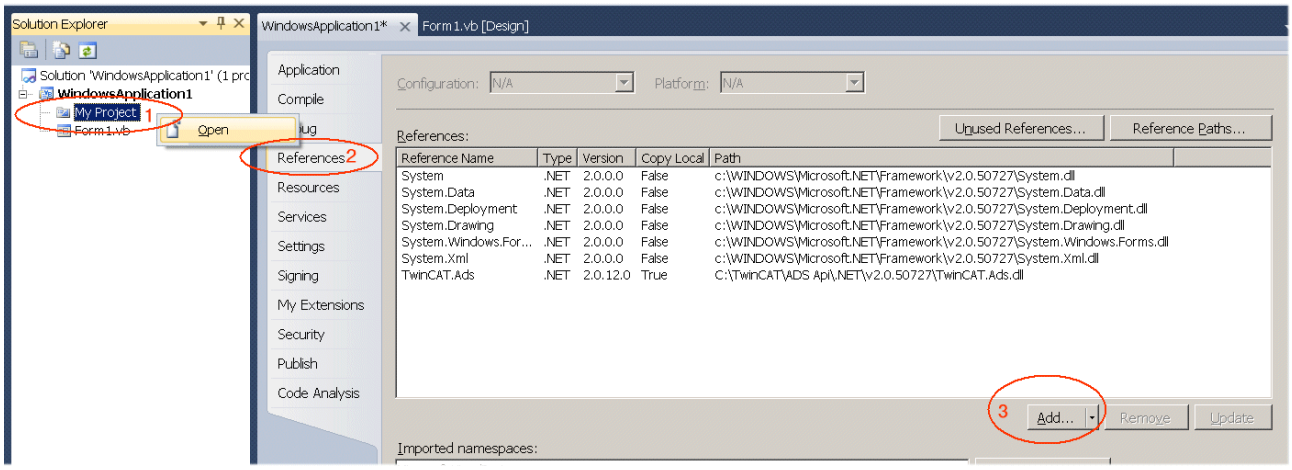
The TwinCAT.Ads component can only be integrated in an existing project. Start Visual Studio and select the following command from the menu: *File->New->Project...*

In the *New Project* dialog please select as project type: *Visual Basic->Windows Forms Application*.

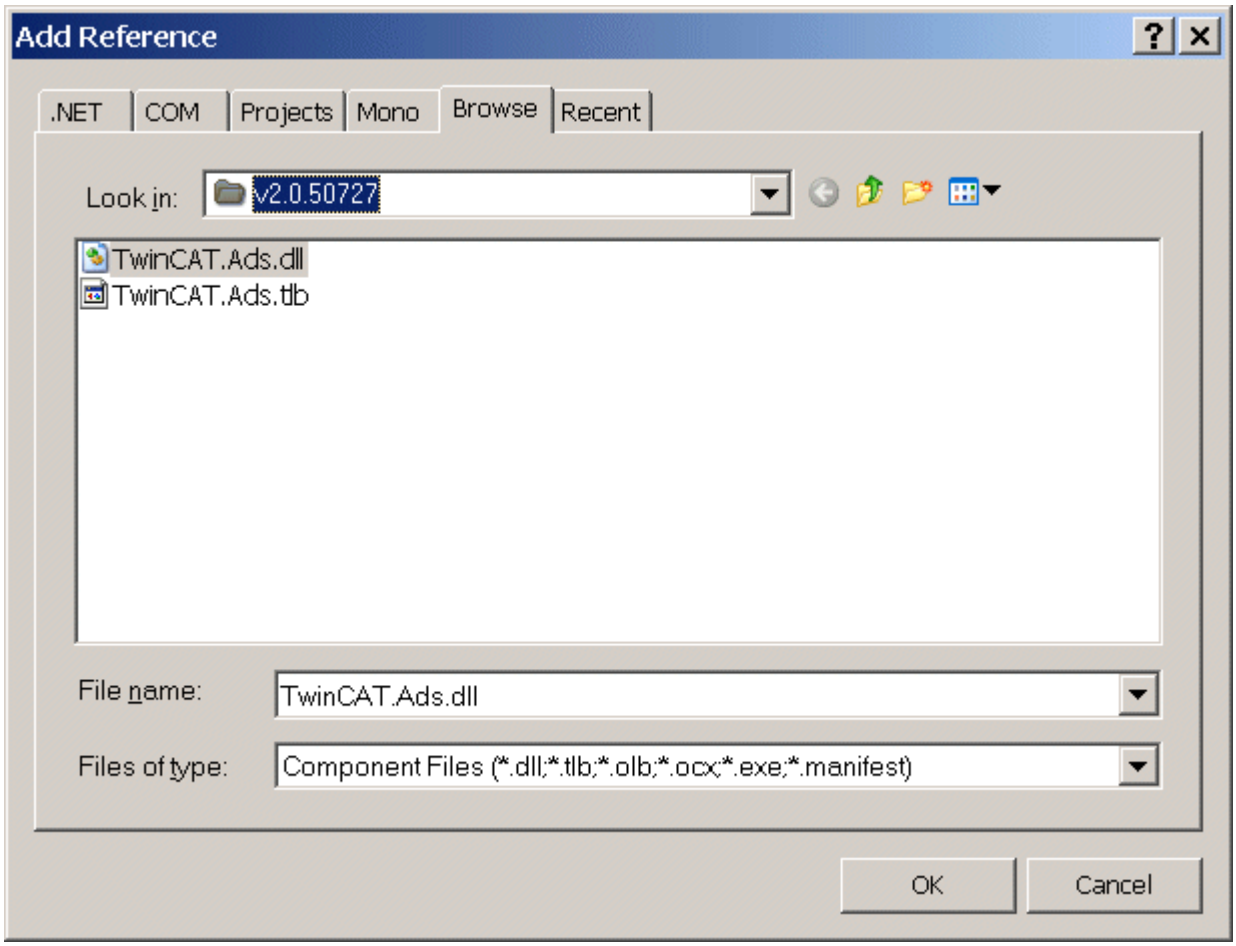


Adding a reference

To select the class library TwinCAT.Ads select the command *Add Reference...* from the *Project* menu. The *Add Reference* dialog opens.
Alternatively you double-click on *My Project* (1) to open the Project Properties window and click the *Add...* button (3) under *Reference (2)*.



In this dialog you should select the *Browse tab* and navigate to file TwinCAT.Ads.dll. Confirm the selected dll with *OK*. By default the TwinCAT.Ads component is located in the ..TwinCAT\Ads Api\.NET\ folder.



Add namespace

All class library types belong to the TwinCAT.Ads namespace. Therefore, the following lines should be added at the beginning of the source text:

```
Imports TwinCAT.Ads
Imports System.IO
```

This enables access to the data types defined in TwinCAT.Ads without having to specify the namespace repeatedly. The TcAdsClient class is the core of the TwinCAT.Ads class library and enables communication with an Ads device. The first step is to create an instance of the TcAdsClient class. Then make use of the Connect method to establish a connection with the ADS device.

Establishing a connection to an Ads device:

```
Private tcAds As TcAdsClient
...

tcAds = New TcAdsClient
tcAds.Connect("1.2.3.4.1.1", 801)

...
```

The data are transmitted with the aid of the AdsStream class, which is derived from System.IO.MemoryStream. The class System.IO.BinaryWriter can be used to write data to the stream, the class System.IO.BinaryReader can be used to read data from the stream.

Reading a PLC variable:

```
' creates a stream with a length of 4 byte
Dim ds = New AdsStream(4)
Dim br = New BinaryReader(ds)

' reads a DINT from PLC
tcAds.Read(&H4020, 0, ds)
ds.Position = 0
TextBox1.Text = br.ReadInt32().ToString()
```

Writing a PLC variable:

```
' creates a stream with a length of 4 byte
Dim ds = New AdsStream(4)
Dim bw = New BinaryWriter(ds)

ds.Position = 0
bw.Write(New Integer = 100)

' writes a DINT to PLC
tcAds.Write(&H4020, 0, ds)
```

Separating a connection to an Ads device:

```
tcAds.Dispose()
```

PLC program:

```
PROGRAM MAIN
VAR
    test AT%MD0 : DINT;
END_VAR
```

```
test := test + 1
```

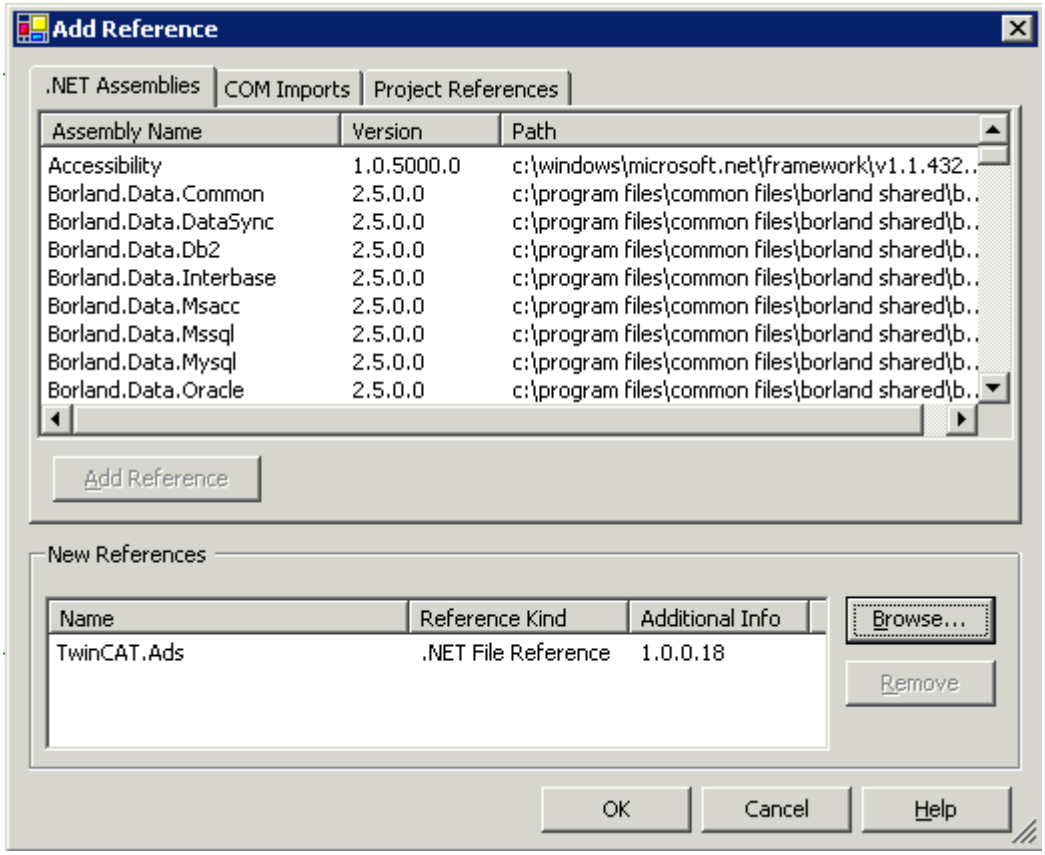
6.2.1.4 Integration in Borland Developer Studio 2006

Creating a new project

Create a new project in Borland Developer Studio (Windows Forms Application).

Add reference

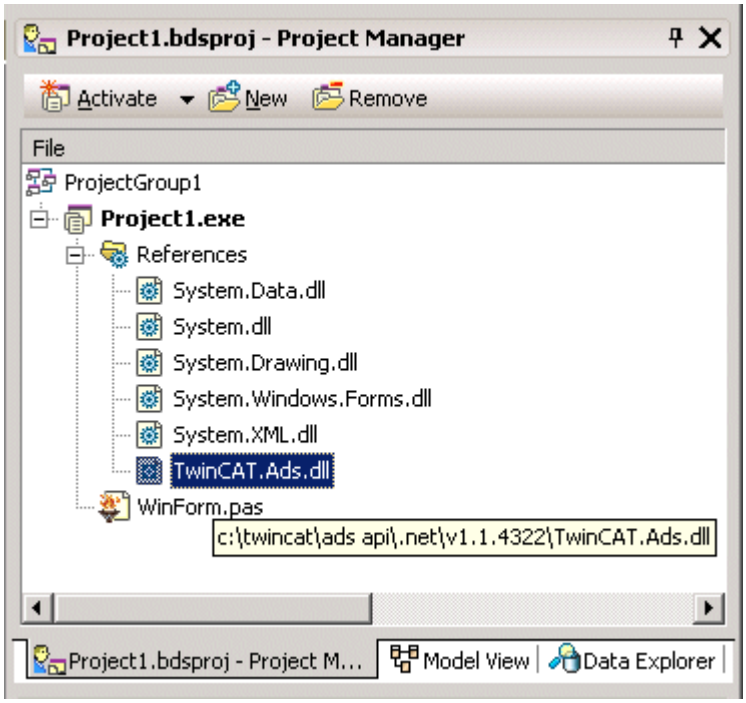
To select the class library TwinCAT.ADS select the command *Add Reference...* from the *Project* menu. The *Add Reference* dialog opens.



In this dialog press the *Browse...* button and select the file TwinCAT.Ads.dll.

Add namespace

The project manager offers an option for checking whether the component was added to the reference list.



All class library types belong to the TwinCAT.ADS namespace. This namespace must be added in the *uses* section:

```
uses
    System.Drawing, System.Collections, System.ComponentModel,
    System.Windows.Forms, System.Data, System.IO, TwinCAT.Ads;
```

This enables access to the data types defined in TwinCAT.ADS without having to specify the namespace repeatedly. The TcAdsClient class is the core of the TwinCAT.ADS class library and enables the user to communicate with an Ads device. The first step is to create an instance of the TcAdsClient class. Then make use of the Connect method to establish a connection with the ADS device.

Establishing a connection to an ADS device:

```
var    tcAds : TcAdsClient;
```

```
procedure TWinForm.Connect();
begin
    tcAds := TcAdsClient.Create();
    tcAds.Connect('172.16.3.217.1.1', 801);
end;
```

Reading a PLC variable:

The data (4 bytes integer in the flag area) are transferred with the aid of the AdsStream class, which is derived from **System.IO.**MemoryStream. The class System.IO.BinaryWriter can be used to write data to the stream, the class System.IO.BinaryReader can be used to read data from the stream.

```
procedure TWinForm.ReadVar();
var     ds : AdsStream;
    br : BinaryReader;
begin
  // creates a stream with a length of 4 byte
 ds := AdsStream.Create(4);
    br := BinaryReader.Create(ds);

  // reads a DINT from PLC
 tcAds.Read($4020,0,ds);

    ds.Position := 0;
    textBox1.Text := br.ReadInt32().ToString();
end;
```

Writing a PLC variable:

```
procedure TWinForm.WriteVar();
var     ds : AdsStream;
    bw : BinaryWriter;
begin
  // creates a stream with a length of 4 byte
 ds := AdsStream.Create(4);
    bw := BinaryWriter.Create(ds);

    ds.Position := 0;
    bw.Write( Int32(100));

  // writes a DINT to PLC
 tcAds.Write($4020,0,ds);
end;
```

Separating a connection to an ADS device:

```
procedure TWinForm.Disconnect();
begin
    tcAds.Dispose();
end;
```

PLC program:

```
PROGRAM MAIN
VAR
    test AT%MD0 : DINT;
END_VAR
```

```
test := test + 1;
```

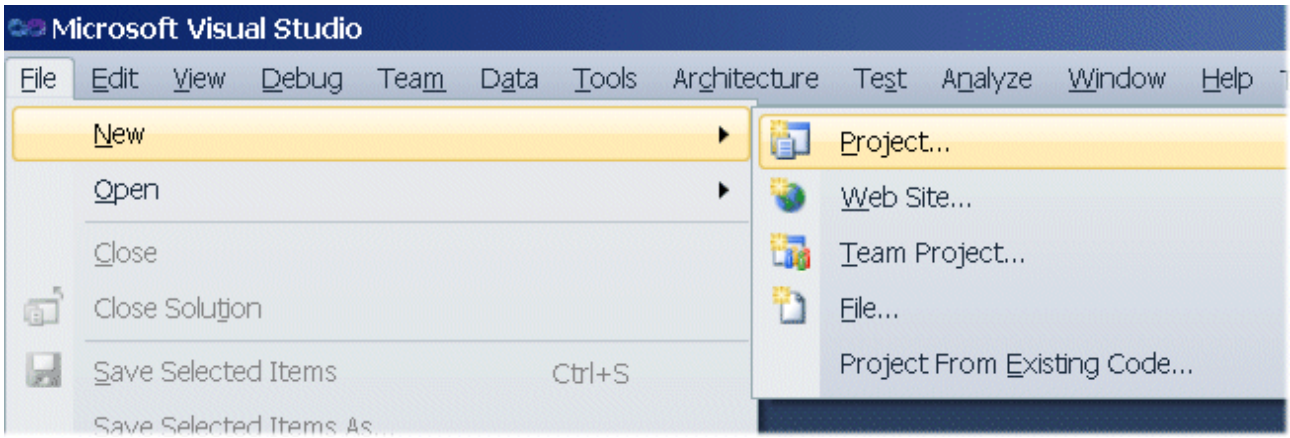
6.2.1.5 Integration in Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)

Embarcadero Prism XE2 = Oxygene for .NET

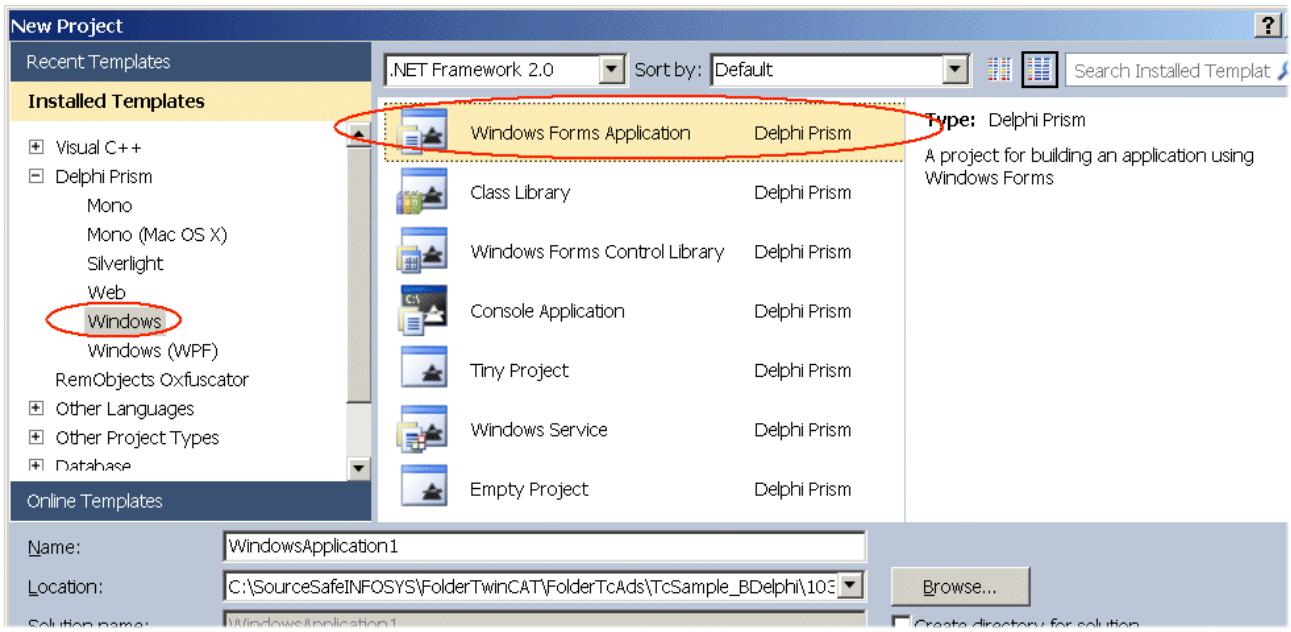
Delphi Prism uses Microsoft Visual Studio 2010 as IDE and integrates itself in an existing VS installation from there. The Visual Studio IDE is supplied with the Delphi Prism installation if it is not yet available.

Create new project

The TwinCAT.ADS component can only be integrated in an existing project. Start VS and select the following command from the menu: *File->New->Project...*

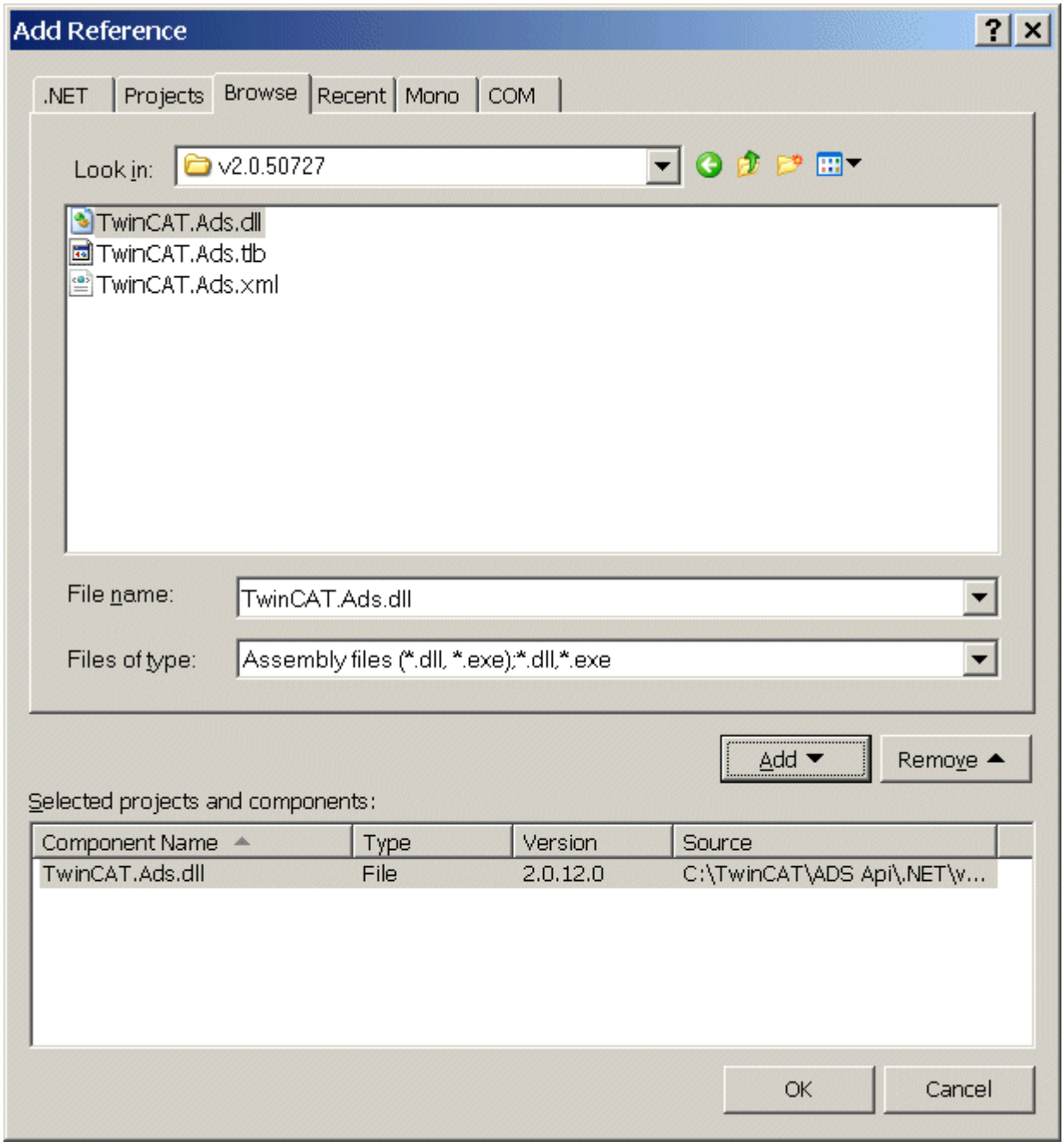


In the *New Project* dialog please select as project type: *Delphi Prism->Windows Forms Application*.



Adding a reference

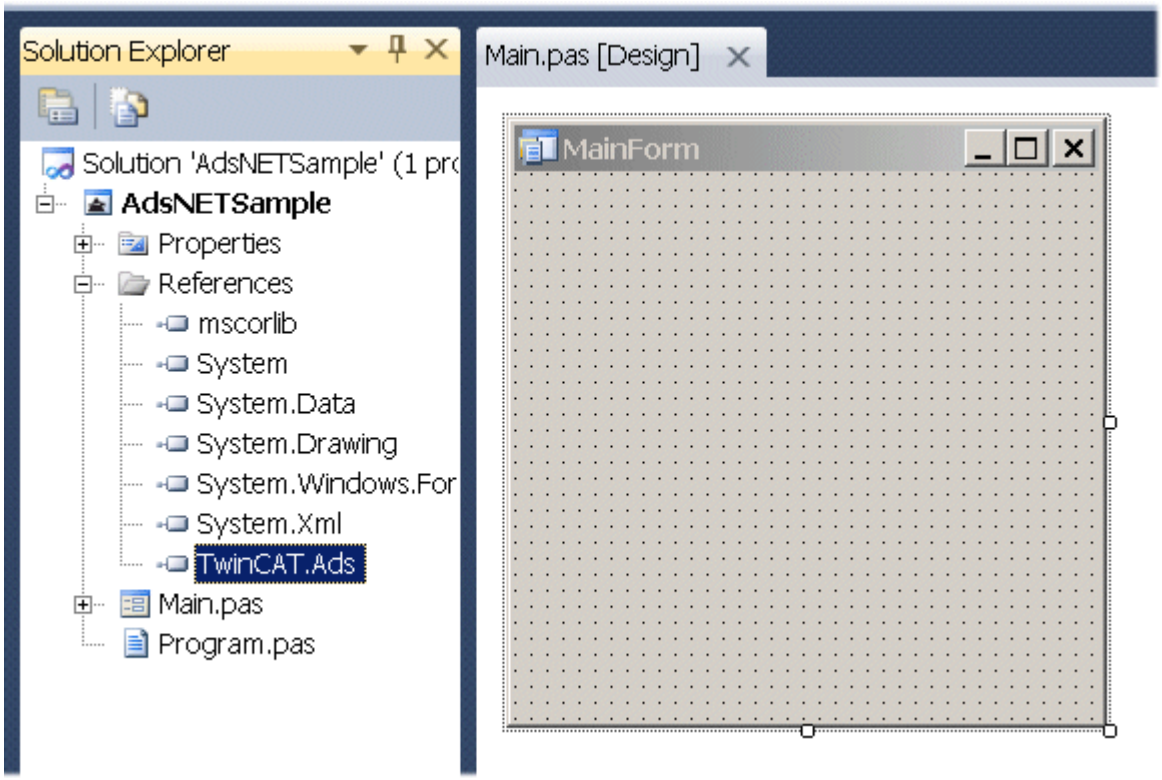
To select the class library TwinCAT.ADS select the command *Add Reference...* from the *Project* menu. The *Add Reference* dialog opens.



In this dialog should select the *Browse tab* and navigate to file TwinCAT.ADS.dll. Confirm the selected dll with *Add* and *OK*. By default the TwinCAT.ADS component is located in folder ..TwinCAT\Ads Api\.NET\ .

Add namespace

The project manager offers an option for checking whether the component was added to the reference list.



All class library types belong to the TwinCAT.ADS namespace. This namespace must be added in the *uses* section:

```
namespace AdsNETSample;'
interface

uses
  System.Drawing,
  System.Collections,
  System.Collections.Generic,
  System.Windows.Forms,
  System.ComponentModel,
  System.IO,TwinCAT.Ads;
```

This enables access to the data types defined in TwinCAT.ADS without having to specify the namespace repeatedly. The TcAdsClient class is the core of the TwinCAT.ADS class library and enables the user to communicate with an ADS device. The first step is to create an instance of the TcAdsClient class. Then make use of the Connect method to establish a connection with the ADS device.

Establishing a connection to an ADS device:

```
type

//...

  MainForm = partial class(System.Windows.Forms.Form)
  private
    tcAds : TcAdsClient;
    method MainForm_Load(sender: System.Object; e: System.EventArgs);
    method MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEventArgs)
;
    method ReadVar_Click(sender: System.Object; e: System.EventArgs);
    method WriteVar_Click(sender: System.Object; e: System.EventArgs);
  protected
    method Dispose(disposing: Boolean); override;
  public
    constructor;
  end;

implementation

//...
```

```
ethod MainForm.MainForm_Load(sender: System.Object; e: System.EventArgs);
begin
    tcAds := new TcAdsClient();
    tcAds.Connect('1.2.3.4.1.1', 801);
end;
```

Reading a PLC variable:

The data (4 bytes integer in the flag area) are transferred with the aid of the AdsStream class, which is derived from **System.IO.**MemoryStream. The class System.IO.BinaryWriter can be used to write data to the stream, the class System.IO.BinaryReader can be used to read data from the stream.

```
method MainForm.ReadVar_Click(sender: System.Object; e: System.EventArgs);
begin
    // creates a stream with a length of 4 byte
 var ds := new AdsStream(4);
    var br := new BinaryReader(ds);
    // reads a DINT from PLC
    tcAds.Read($4020,0,ds);
    ds.Position := 0;
    textBox1.Text := br.ReadInt32().ToString();
end;
```

Writing a PLC variable:

```
method MainForm.WriteVar_Click(sender: System.Object; e: System.EventArgs);
begin
    // creates a stream with a length of 4 byte
 var ds := new AdsStream(4);
    var bw := new BinaryWriter(ds);
    ds.Position := 0;
    bw.Write( Int32(100));
    // writes a DINT to PLC
    tcAds.Write($4020,0,ds);
end;
```

Separating a connection to an ADS device:

```
method MainForm.MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEvent
Args);
begin
    tcAds.Dispose();
end;
```

PLC program:

```
PROGRAM MAIN
VAR
    test AT%MD0 : DINT;
END_VAR
```

```
test := test + 1;
```

6.2.2 Accessing an array in the PLC

Download

Language / IDE	Unpack the example program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490113291/.zip
Visual Basic (for .NET Framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490114699/.exe
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490116107/.exe

Language / IDE	Unpack the example program
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490117515/.exe

Task

The PLC contains an array that is to be read by the .Net application using a read command.

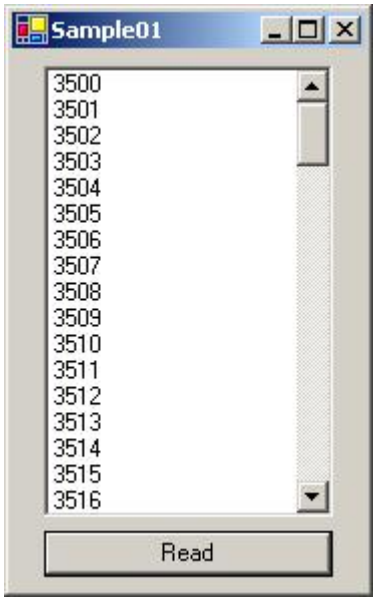
Description

The PLC contains an array of 100 elements of type integer (2 bytes). The array in the PLC is to be filled with the values from 3500 to 3599.

In the Form1_Load event method a new instance of the class TcAdsClient is created. Then the method TcAdsClient.Connec [▶ 689]t of the TcAdsClient object is called to establish a connection to the port 801. Finally the method TcAdsClient.CreateVariableHandle [▶ 696] is used to fetch the handle of the PLC variable. When the program finishes, this handle is released in the Form1_Closing event method and the Dispose method of the TcAdsClient object is called.

When the user clicks the button on the form, the entire array is read from the PLC into the AdsStream dataStream by means of the TcAdsClient.Read [▶ 708] method. The stream should be of the same size as the data in the SPS. Because we want to read 100 INTs (each 2 Bytes), the stream must be able to hold at least 2 * 100 Bytes.

The class System.IO.BinaryReader is necessary to read the individual fields of the array. First the position of the stream has to be set to 0. Then the individual fields of the array can be read by calling the method BinaryReader.ReadInt16 in a for loop. The position pointer of the stream is incremented automatically by the bytes (in this case 2) that have been read.



C# program

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using TwinCAT.Ads;
using System.IO;

namespace Sample01
{
    public class Form1 : System.Windows.Forms.Form
```

```
    {
        private System.Windows.Forms.Button btnRead;
        private System.ComponentModel.Container components = null;
        private System.Windows.Forms.ListBox lbArray;

        private int hVar;
        private TcAdsClient tcClient;

        public Form1()
        {
            InitializeComponent();
        }

        protected override void Dispose( bool disposing ) ...
        private void InitializeComponent() ...

        [STAThread]
        static void Main()
        {
            Application.Run(new Form1());
        }

        private void Form1_Load(object sender, System.EventArgs e)
        {
            // Create instance of class TcAdsClient
            tcClient = new TcAdsClient();


            // Connect to local PLC - Runtime 1 - TwinCAT2 Port=801, TwinCAT3 Port=851
            tcClient.Connect(801);

            try
            {
                hVar = tcClient.CreateVariableHandle("MAIN.PLCVar");
            }
            catch(Exception err)
            {
                MessageBox.Show(err.Message);
            }
        }

        private void btnRead_Click(object sender, System.EventArgs e)
        {
            try
            {
                // AdsStream which gets the data
                AdsStream dataStream = new AdsStream(100 * 2);
                BinaryReader binRead = new BinaryReader(dataStream);

                //read comlpete Array
                tcClient.Read(hVar,dataStream);

                lbArray.Items.Clear();
                dataStream.Position = 0;
                for(int i=0; i<100; i++)
                {
                    lbArray.Items.Add(binRead.ReadInt16().ToString());
                }
            }
            catch(Exception err)
            {
                MessageBox.Show(err.Message);
            }
        }

        private void Form1_Closing(object sender, System.ComponentModel.CancelEventArgs e)
        {
            //enable resources
            try
            {
                tcClient.DeleteVariableHandle(hVar);
            }
            catch(Exception err)
            {
                MessageBox.Show(err.Message);
            }
            tcClient.Dispose();
        }
    }
}
```

PLC program

```
PROGRAM MAIN
VAR
  PLCVar : ARRAY [0..99] OF INT;
  Index: BYTE;
END_VAR

FOR Index := 0 TO 99 DO
  PLCVar[Index] := 3500 + INDEX;
END_FOR
```

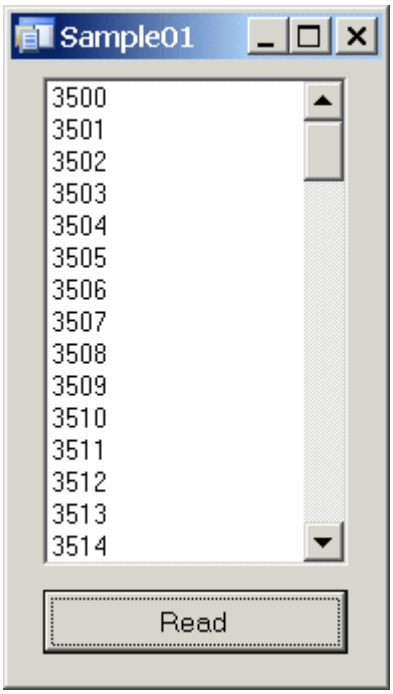
6.2.2.1 Accessing an array in the PLC with Delphi

Task

The PLC contains an array that is to be read by the .Net application using a read command.

Description

The PLC contains an array of 100 elements of type integer (2 bytes). The array in the PLC is to be filled with the values from 3500 to 3599.



Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample01;

interface

uses
  System.Drawing,
  System.Collections,
  System.Collections.Generic,
  System.Windows.Forms,
  System.ComponentModel, System.IO, TwinCAT.Ads;

type
  /// <summary>
```

```
  /// Summary description for MainForm.
  /// </summary>
  MainForm = partial class(System.Windows.Forms.Form)
  private
    method MainForm_Load(sender: System.Object; e: System.EventArgs);
    method MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEventArgs)
;
    method btnRead_Click(sender: System.Object; e: System.EventArgs);
    tcClient  : TcAdsClient;
    hVar      : Int32;
  protected
    method Dispose(disposing: Boolean); override;
  public
    // PLCVar : ARRAY [0..99] OF INT; array of 2 byte signed integer
    const MAX_ARRAY_ELEMENTS = 100;// Max. number of PLC array elements
    const MAX_ADSSTREAM_SIZE = MAX_ARRAY_ELEMENTS * 2;// Max. byte length of array data
    constructor;
  end;

implementation

{$REGION Construction and Disposition}
constructor MainForm;
begin
  //
  // Required for Windows Form Designer support
  //
  InitializeComponent();

  //
  // TODO: Add any constructor code after InitializeComponent call
  //
end;

method MainForm.Dispose(disposing: Boolean);
begin
  if disposing then begin
    if assigned(components) then
      components.Dispose();

    //
    // TODO: Add custom disposition code here
    //
  end;
  inherited Dispose(disposing);
end;
{$ENDREGION}

method MainForm.MainForm_Load(sender: System.Object; e: System.EventArgs);
begin
  try
    // Create instance of class TcAdsClient
    tcClient := new TcAdsClient();
    // Create connection with Port 801 on local PC
    tcClient.Connect(801);
    // Create PLC variable handle
    hVar := tcClient.CreateVariableHandle('MAIN.PLCVar')
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

method MainForm.MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEvent
Args);
begin
  try
    // Release PLC variable handle
    tcClient.DeleteVariableHandle(hVar)
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
  // Close connection
  tcClient.Dispose();
end;

method MainForm.btnRead_Click(sender: System.Object; e: System.EventArgs);
begin
  try
```

```
    // Create stream which gets the data
    var dataStream := new AdsStream(MAX_ADSSTREAM_SIZE);
    // Create BinaryReader
    var binRead := new BinaryReader(dataStream);
    // Read complete array from PLC to stream
    tcClient.Read(hVar,dataStream);
    // Clear list box
    lbArray.Items.Clear();
    // Reset stream position
    dataStream.Position := 0;
    // Read all data from stream, convert to string and add list box items
    for I:Integer := 1 to MAX_ARRAY_ELEMENTS do
      lbArray.Items.Add(binRead.ReadInt16().ToString());
  except
      on err: Exception do
    MessageBox.Show(err.Message, err.Source)
  end;
end;
```

PLC program

```
PROGRAM MAIN
VAR
    PLCVar : ARRAY [0..99] OF INT;
    Index: BYTE;
END_VAR
```

```
FOR Index := 0 TO 99 DO
    PLCVar[Index] := 3500 + INDEX;
END_FOR
```

Download

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466754699/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466756107/.exe

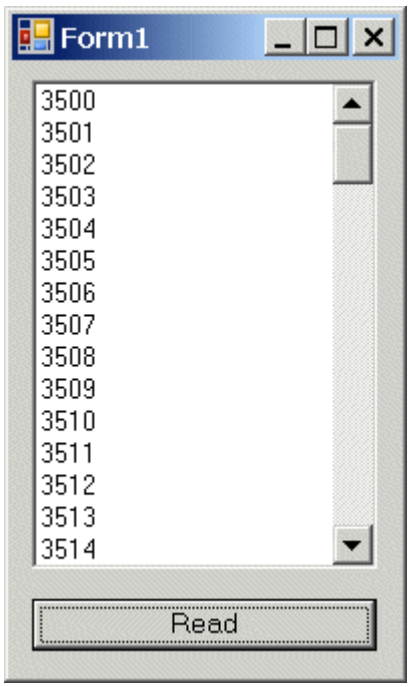
6.2.2.2 Accessing an array in the PLC with Visual Basic

Task

The PLC contains an array that is to be read by the .Net application using a read command.

Description

The PLC contains an array of 100 elements of type integer (2 bytes). The array in the PLC is to be filled with the values from 3500 to 3599.



Visual Basic (for .NET framework) program

```
Imports TwinCAT.Ads
Imports System.IO

Public Class Form1
    Inherits System.Windows.Forms.Form

#Region " Windows Form Designer generated code "

...

#End Region

    Dim hVar As Integer
    Dim tcClient As TcAdsClient

    Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles MyBase.Load
    tcClient = New TcAdsClient()
    tcClient.Connect(801)

    Try
        hVar = tcClient.CreateVariableHandle("MAIN.PLCVar")
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try

    End Sub

    Private Sub Form1_Closing(ByVal sender As Object, ByVal e As System.ComponentModel.CancelEventAr
gs) Handles MyBase.Closing
    'Release resources
    Try
        tcClient.DeleteVariableHandle(hVar)
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    tcClient.Dispose()
    End Sub

    Private Sub btnRead_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles bt
nRead.Click
    Try
        Dim dataStream As New AdsStream(100 * 2)
        Dim binRead As New BinaryReader(dataStream)

        'Read complete array
        tcClient.Read(hVar, dataStream)
```

```
        lbArray.Items.Clear()
        dataStream.Position = 0
        Dim i As Integer
        For i = 0 To 99
        lbArray.Items.Add(binRead.ReadInt16().ToString())
        Next
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub

End Class
```

PLC program

```
PROGRAM MAIN
VAR
    PLCVar : ARRAY [0..99] OF INT;
    Index: BYTE;
END_VAR
```

```
FOR Index := 0 TO 99 DO
    PLCVar[Index] := 3500 + INDEX;
END_FOR
```

Download

Language / IDE	Unpack the sample program
Visual Basic (for .NET framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12463809291/.exe

6.2.3 Transmitting structures to the PLC

Download

Language / IDE	Unpack the example program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490118923/.zip
Visual Basic (for .NET Framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490120331/.exe
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490121739/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490123147/.exe

Task

A structure is to be written into the PLC by the .NET application. The elements in the structure have different data types.

Description

The structure we want to write into the PLC:

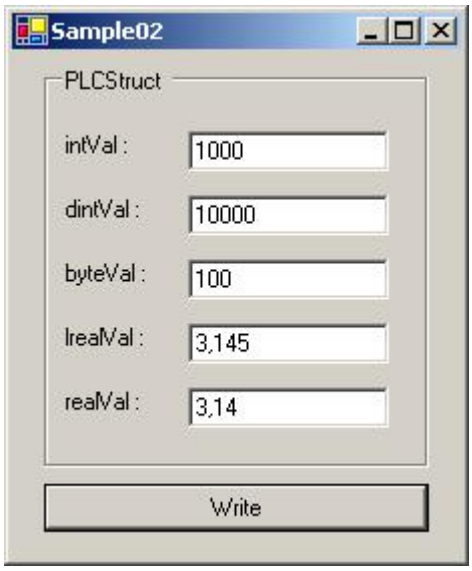
```
TYPE PLCStruct
STRUCT
  intVal   : INT;  (*Offset 0*)
```

```
  dintVal  : DINT; (*Offset 2*)
  byteVal  : SINT; (*Offset 6*)
  lrealVal : LREAL;(*Offset 7*)
  realVal  : REAL; (*Offset 15 --> Total size 19 Bytes*)
END_STRUCT
END_TYPE
```

In c# the structure would look like this:

```
public struct PLCStruct
{
  public short  intVal;
  public int    dintVal;
  public byte   byteVal;
  public double     lrealVal;
  public float  realVal;
}
```

This structure will not be created directly. Instead we will use a AdsStream as in Sample01. This object is initialized with a size of 19 bytes in the write event method. This is equivalent to the size of the data type PLCStruct in the SPS. A BinaryWriter is used to write the data from the textboxes to the stream. After setting the stream position to 0 the individual fields are written to the stream. Finally the total stream is written into the SPS with the help of TcAdsClient.Write [▶ 754].



C# Program

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using System.IO;
using TwinCAT.Ads;

namespace Sample02
{
    ///
    /// Summary description for Form1.
    ///
    public class Form1 : System.Windows.Forms.Form
    {
        private System.Windows.Forms.GroupBox groupBox1;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.Label label5;
```

```
        private System.Windows.Forms.Button btnWrite;
        private System.Windows.Forms.TextBox tbInt;
        private System.Windows.Forms.TextBox tbDint;
        private System.Windows.Forms.TextBox tbByte;
        private System.Windows.Forms.TextBox tbLReal;
        private System.Windows.Forms.TextBox tbReal;

        private System.ComponentModel.Container components = null;

        private int hVar;private TcAdsClient tcClient;

        public Form1()
        {
            InitializeComponent();
        }


        protected override void Dispose( bool disposing )...

        private void InitializeComponent()...

        [STAThread]
        static void Main()
        {
            Application.Run(new Form1());
        }

        private void Form1_Load(object sender, System.EventArgs e)
        {
            // Create instance of class TcAdsClient
            tcClient = new TcAdsClient();

            // Connect to local PLC - Runtime 1 - TwinCAT2 Port=801, TwinCAT3 Port=851
            tcClient.Connect(801);

            try
            {
                hVar = tcClient.CreateVariableHandle("MAIN.PLCVar");
            }
            catch(Exception err)
            {
                MessageBox.Show(err.Message);
            }
        }

        private void btnWrite_Click(object sender, System.EventArgs e)
        {
            AdsStream dataStream = new AdsStream(19);
            BinaryWriter binWrite = new BinaryWriter(dataStream);

            dataStream.Position = 0;
            try
            {
                //Fill stream according to the order with data from the text boxes
                binWrite.Write(short.Parse(tbInt.Text));
                binWrite.Write(int.Parse(tbDint.Text));
                binWrite.Write(byte.Parse(tbByte.Text));
                binWrite.Write(double.Parse(tbLReal.Text));
                binWrite.Write(float.Parse(tbReal.Text));

                //Write complete stream in the PLC
                tcClient.Write(hVar,dataStream);
            }
            catch( Exception err)
            {
                MessageBox.Show(err.Message);
            }
        }

        private void Form1_Closing(object sender, System.ComponentModel.CancelEventArgs e)
        {
            //Enable resources
            try
            {
                tcClient.DeleteVariableHandle(hVar);
            }
            catch(Exception err)
            {
                MessageBox.Show(err.Message);
            }
```

```
            tcClient.Dispose();
        }
    }
}
```

PLC program

```
TYPE PLCStruct
STRUCT
  intVal : INT;
  dintVal    : DINT;
  byteVal    : SINT;
  lrealVal  : LREAL;
  realVal  : REAL;
END_STRUCT
END_TYPE

PROGRAM MAIN
VAR
  PLCVar : PLCStruct;
END_VAR
```

6.2.3.1 Transmitting a structure to the PLC with Delphi

Task

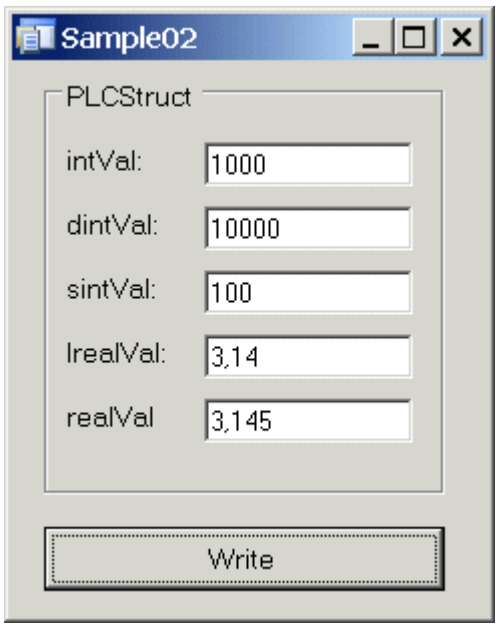
A structure is to be written into the PLC by the .NET application. The elements in the structure have different data types.

Description

The structure we want to write into the PLC:

```
TYPE PLCStruct
STRUCT
  intVal   : INT;  (*Offset 0*)
  dintVal  : DINT; (*Offset 2*)
  byteVal  : SINT; (*Offset 6*)
  lrealVal : LREAL;(*Offset 7*)
  realVal  : REAL; (*Offset 15 --> Total size 19 Bytes*)
END_STRUCT
END_TYPE
```

The AdsStream object is initialized with a size of 19 bytes in the write event method. This is equivalent to the size of the data type PLCStruct in the SPS. A BinaryWriter is used to write the data from the textboxes to the stream. After setting the stream position to 0 the individual fields are written to the stream. Finally, the total stream is written into the SPS with the help of TcAdsClient.Write [▶ 754].



Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample02;

interface

uses
  System.Drawing,
  System.Collections,
  System.Collections.Generic,
  System.Windows.Forms,
  System.ComponentModel, System.IO, TwinCAT.Ads;

type
  /// <summary>
  /// Summary description for MainForm.
  /// </summary>
  MainForm = partial class(System.Windows.Forms.Form)
  private
    method btnWrite_Click(sender: System.Object; e: System.EventArgs);
    method MainForm_Load(sender: System.Object; e: System.EventArgs);
    method MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEventArgs)
;
    tcClient  : TcAdsClient;
    hVar      : Integer;
  protected
    method Dispose(disposing: Boolean); override;
  public
    // TYPE PLCStruct :
    // STRUCT
    //    intVal    : INT; 2 byte, signed integer
    //    dintVal       : DINT; 4 byte, signed integer
    //    sintVal       : SINT; 1 byte, signed integer
    //    lrealVal: LREAL; 8 byte, floating point value
    //    realVal       : REAL; 4 byte, floating point value
    // END_STRUCT
    // END_TYPE
    // Length of all PLC structure members = 19 byte (x86, byte alignment)
    const MAX_ADSSTREAM_SIZE = 19;// SIZEOF(PLCStruct), Max. byte length of PLC structure
    constructor;
  end;

implementation

{$REGION Construction and Disposition}
constructor MainForm;
begin
  //
  // Required for Windows Form Designer support
  //
  InitializeComponent();
```

```
  //
  // TODO: Add any constructor code after InitializeComponent call
  //
end;

method MainForm.Dispose(disposing: Boolean);
begin
  if disposing then begin
    if assigned(components) then
      components.Dispose();

    //
    // TODO: Add custom disposition code here
    //
  end;
  inherited Dispose(disposing);
end;
{$ENDREGION}

method MainForm.btnWrite_Click(sender: System.Object; e: System.EventArgs);
begin
  try
    // Create stream which holds the data
    var dataStream := new AdsStream(MAX_ADSSTREAM_SIZE);
    // Create BinaryWriter
    var binWrite := new BinaryWriter(dataStream);
    // Reset stream to start position
    dataStream.Position := 0;
    // Write structure member data to the stream
    binWrite.Write(System.Int16.Parse(tbInt.Text));
    binWrite.Write(System.Int32.Parse(tbDint.Text));
    binWrite.Write(System.SByte.Parse(tbSInt.Text));
    binWrite.Write(System.Double.Parse(tbLReal.Text));
    binWrite.Write(System.Single.Parse(tbReal.Text));
    // Write stream to the PLC
    tcClient.Write(hVar,dataStream)
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

method MainForm.MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEvent
Args);
begin
  try
    // Release PLC variable handle
    tcClient.DeleteVariableHandle(hVar)
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
  // Close connection
  tcClient.Dispose();
end;

method MainForm.MainForm_Load(sender: System.Object; e: System.EventArgs);
begin
  try
    // Create instance of class TcAdsClient
    tcClient := new TcAdsClient();
    // Create connection with Port 801 on local PC
    tcClient.Connect(801);
    // Create PLC variable handle
    hVar := tcClient.CreateVariableHandle('MAIN.PLCVar')
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

end.
```

PLC program

```
TYPE PLCStruct :
STRUCT
    intVal : INT;
```

```
    dintVal : DINT;
    sintVal : SINT;
    lrealVal : LREAL;
    realVal : REAL;
END_STRUCT
END_TYPE
```

```
PROGRAM MAIN
VAR
    PLCVar:PLCStruct;
END_VAR
```

Requirements

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466757515/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466758923/.exe

6.2.3.2 Transmitting a structure to the PLC with Visual Basic

Task

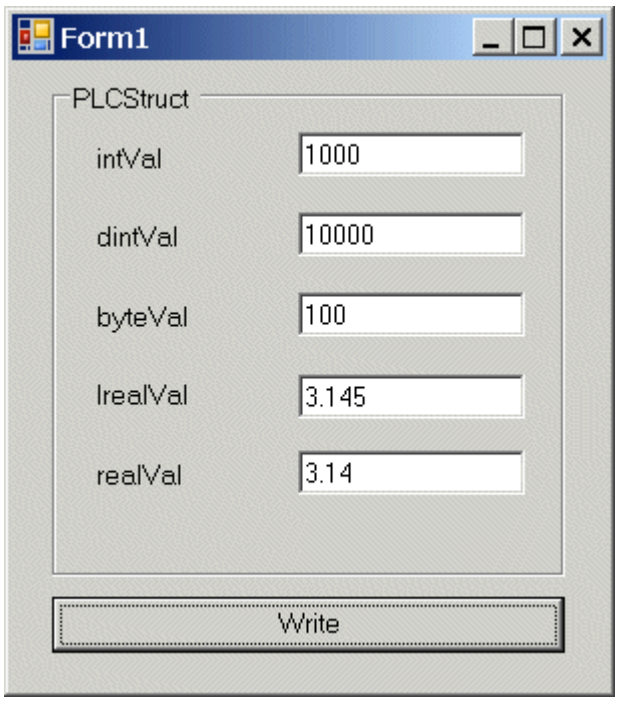
A structure is to be written into the PLC by the .NET application. The elements in the structure have different data types.

Description

The structure we want to write into the PLC:

```
TYPE PLCStruct
STRUCT
  intVal   : INT;  (*Offset 0*)
  dintVal  : DINT; (*Offset 2*)
  byteVal  : SINT; (*Offset 6*)
  lrealVal : LREAL;(*Offset 7*)
  realVal  : REAL; (*Offset 15 --> Total size 19 Bytes*)
END_STRUCT
END_TYPE
```

The AdsStream object is initialized with a size of 19 bytes in the write event method. This is equivalent to the size of the data type PLCStruct in the SPS. A BinaryWriter is used to write the data from the textboxes to the stream. After setting the stream position to 0 the individual fields are written to the stream. Finally the total stream is written into the SPS with the help of TcAdsClient.Write [▶ 754].



Visual Basic (for .NET framework) program

```
Imports TwinCAT.Ads
Imports System.IO

Public Class Form1
    Inherits System.Windows.Forms.Form

#Region " Windows Form Designer generated code "

...

#End Region

    Dim hVar As Integer
    Dim tcClient As TcAdsClient

    Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles MyBase.Load
    ' Create instance of TcAdsClient
    tcClient = New TcAdsClient()

    ' Establish connection to local system port 801
    tcClient.Connect(801)

    Try
        hVar = tcClient.CreateVariableHandle("MAIN.PLCVar")
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub

    Private Sub Form1_Closing(ByVal sender As Object, ByVal e As System.ComponentModel.CancelEventAr
gs) Handles MyBase.Closing
    ' Release resources
    Try
        tcClient.DeleteVariableHandle(hVar)
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    tcClient.Dispose()
    End Sub


    Private Sub btnWrite_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnWrite
.Click
    Dim dataStream As New AdsStream(19)
    Dim binWrite As New BinaryWriter(dataStream)
```

```
    dataStream.Position = 0

    Try
        binWrite.Write(Short.Parse(tbInt.Text))
        binWrite.Write(Integer.Parse(tbDint.Text))
        binWrite.Write(Byte.Parse(tbByte.Text))
        binWrite.Write(Double.Parse(tbLReal.Text))
        binWrite.Write(Single.Parse(tbReal.Text))

        tcClient.Write(hVar, dataStream)

    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub
End Class
```

PLC program

```
TYPE PLCStruct :
STRUCT
    intVal : INT;
    dintVal : DINT;
    sintVal : SINT;
    lrealVal : LREAL;
    realVal : REAL;
END_STRUCT
END_TYPE
```

```
PROGRAM MAIN
VAR
    PLCVar:PLCStruct;
END_VAR
```

Download

Language / IDE	Unpack the sample program
Visual Basic (for .NET framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12463810699/.exe

6.2.4 Event driven reading

Download

Language / IDE	Unpack the example program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490124555/.zip
Visual Basic (for .NET Framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490125963/.exe
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490127371/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490128779/.exe

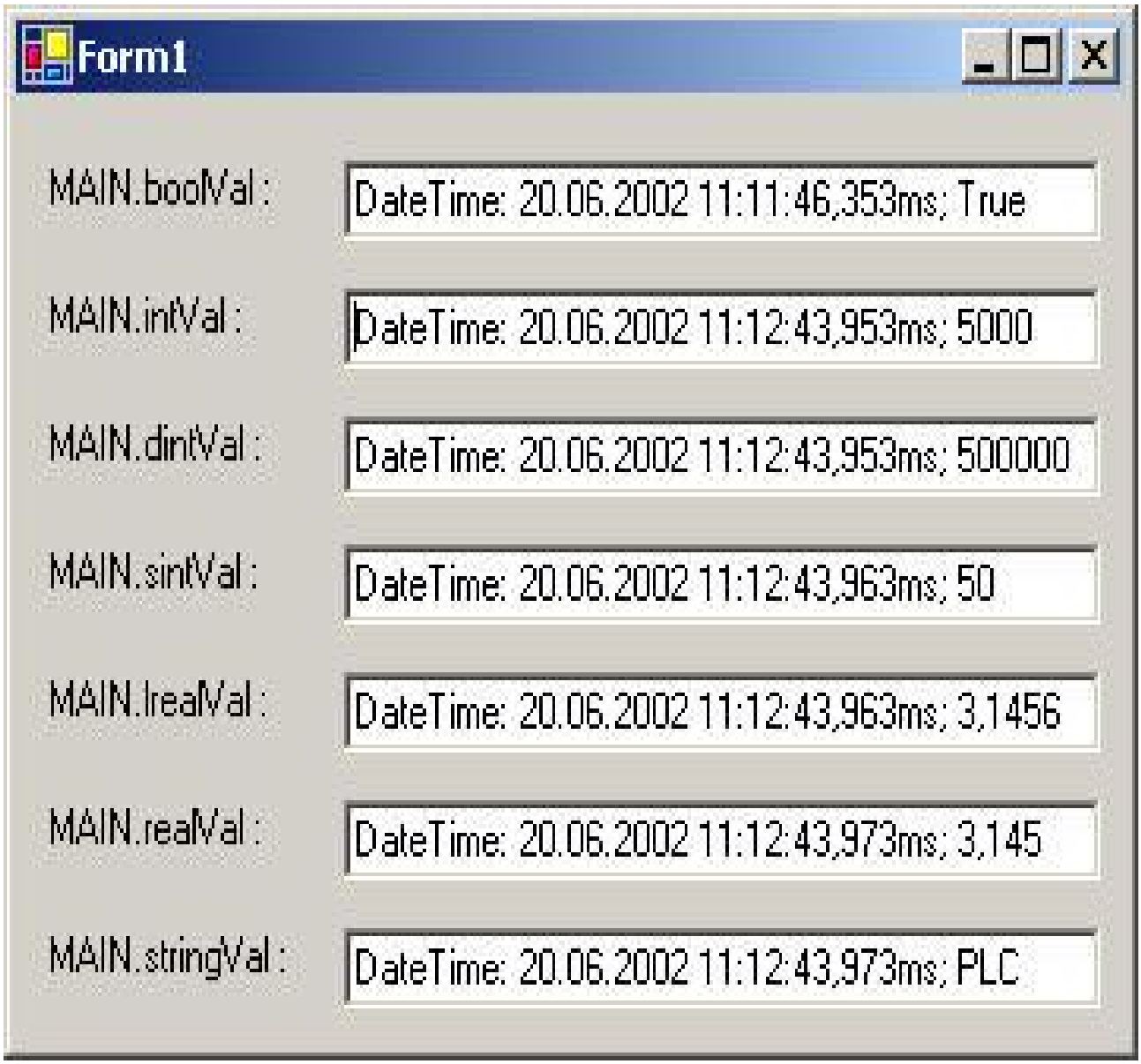
Task

There are 7 global variables in the PLC. Each of these PLC variables is of a different data type. The values of the variables should be read in the most effective manner, and the value with its timestamp is to be displayed on a form.

Description

In the form's load event, a connection to each of the PLC variables is created with the TcAdsClient.AddDeviceNotification() [▶ 657] method. The handle for this connection is stored in a array. The parameter TransMode specifies the type of data exchange. AdsTransMode.OnChange has been selected here. This means that the value of the PLC variable is only transmitted if its value within the PLC has changed (see the AdsTransMode [▶ 401] data type). The parameter cycleTime indicates that the PLC is to check whether the corresponding variable has changed every 100 ms. MaxDelay allows to collect notification for a specified interval. If the maxDelay elapse, all notifications will be send at once.
When the PLC variable changes, the TcAdsClient.AdsNotification() [▶ 774] event is called. The parameter e of the event handling method is of the type AdsNotificationEventArgs and contains the time stamp, the handle, the value and the control in which the value is to be displayed. The connections are released again in the closing event by means of the TcAdsClient.DeleteDeviceNotification() [▶ 697] method. It is essential that you do this, since every connection established by TcAdsClient.AddDeviceNotification() uses resources.
You should also choose appropriate values for the cycle time, since too many write / read operations load the system so heavily that the user interface becomes much slower.

Advice: Don't use time intensive executions in callbacks (OnNotification()).



C# program

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using System.IO;
using TwinCAT.Ads;

namespace Sample03
{
    public class Form1 : System.Windows.Forms.Form
    {
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.Label label5;
        private System.Windows.Forms.Label label6;
        private System.Windows.Forms.Label label7;
        private System.Windows.Forms.Label label8;
        private System.Windows.Forms.Label label9;
        private System.Windows.Forms.Label label10;
        private System.Windows.Forms.TextBox tbInt;
        private System.Windows.Forms.TextBox tbDint;
        private System.Windows.Forms.TextBox tbSint;
        private System.Windows.Forms.TextBox tbLreal;
        private System.Windows.Forms.TextBox tbReal;
        private System.Windows.Forms.TextBox tbString;
        private System.Windows.Forms.Label label11;
        private System.Windows.Forms.TextBox tbBool;
        private System.ComponentModel.Container components = null;

        private TcAdsClient tcClient;
        private int[] hConnect;
        private AdsStream dataStream;
        private BinaryReader binRead;

        public Form1()
        {
            InitializeComponent();
        }

        protected override void Dispose( bool disposing ) ...

        private void InitializeComponent() ...

        [STAThread]
        static void Main()
        {
            Application.Run(new Form1());
        }

        private void Form1_Load(object sender, System.EventArgs e)
        {
            dataStream = new AdsStream(31);
            //Encoding is set to ASCII, to read strings
            binRead = new BinaryReader(dataStream, System.Text.Encoding.ASCII);
            // Creaste instance of class TcAdsClient
            tcClient = new TcAdsClient();

            // PLC1 Port: TwinCAT2=801, TwinCAT3=851
            tcClient.Connect(801);

            hConnect = new int[7];

            try
            {
                hConnect[0] = tcClient.AddDeviceNotification("MAIN.boolVal",dataStream,0,1,
                    AdsTransMode.OnChange,100,0,tbBool);
                hConnect[1] = tcClient.AddDeviceNotification("MAIN.intVal",dataStream,1,2,
                    AdsTransMode.OnChange,100,0,tbInt);
                hConnect[2] = tcClient.AddDeviceNotification("MAIN.dintVal",dataStream,3,4,
                    AdsTransMode.OnChange,100,0,tbDint);
                hConnect[3] = tcClient.AddDeviceNotification("MAIN.sintVal",dataStream,7,1,
                    AdsTransMode.OnChange,100,0,tbSint);
                hConnect[4] = tcClient.AddDeviceNotification("MAIN.lrealVal",dataStream,8,8,
                    AdsTransMode.OnChange,100,0,tbLreal);
```

```
                hConnect[5] = tcClient.AddDeviceNotification("MAIN.realVal",dataStream,16,4,
                    AdsTransMode.OnChange,100,0,tbReal);
                hConnect[6] = tcClient.AddDeviceNotification("MAIN.stringVal",dataStream,20,11,
                    AdsTransMode.OnChange,100,0,tbString);

                tcClient.AdsNotification += new AdsNotificationEventHandler(OnNotification);
            }
            catch(Exception err)
            {
                MessageBox.Show(err.Message);
            }
        }

        private void OnNotification(object sender, AdsNotificationEventArgs e)
        {
            DateTime time = DateTime.FromFileTime(e.TimeStamp);
            e.DataStream.Position = e.Offset;
            string strValue = "";

            if( e.NotificationHandle == hConnect[0])
                strValue = binRead.ReadBoolean().ToString();
            else if( e.NotificationHandle == hConnect[1] )
                strValue = binRead.ReadInt16().ToString();
            else if( e.NotificationHandle == hConnect[2] )
                strValue = binRead.ReadInt32().ToString();
            else if( e.NotificationHandle == hConnect[3] )
                strValue = binRead.ReadSByte().ToString();
            else if( e.NotificationHandle == hConnect[4] )
                strValue = binRead.ReadDouble().ToString();
            else if( e.NotificationHandle == hConnect[5] )
                strValue = binRead.ReadSingle().ToString();
            else if( e.NotificationHandle == hConnect[6] )
            {
                strValue = new String(binRead.ReadChars(11));
            }

            ((TextBox)e.UserData).Text = String.Format("DateTime: {0},{1}
ms; {2}",time,time.Millisecond,strValue);
        }

        private void Form1_Closing(object sender, System.ComponentModel.CancelEventArgs e)
        {
            try
            {
                for(int i=0; i<7; i++)
                {
                    tcClient.DeleteDeviceNotification(hConnect[i]);
                }
            }
            catch(Exception err)
            {
                MessageBox.Show(err.Message);
            }
            tcClient.Dispose();
        }

    }
}
```

PLC program

```
PROGRAM MAIN
VAR
 boolVal      : BOOL;
 intVal    : INT;
 dintVal   : DINT;
 sintVal   : SINT;
 lrealVal  : LREAL;
 realVal   : REAL;
 stringVal : STRING(10);
END_VAR

PROGRAM MAIN
VAR
    ;
END_VAR
```

6.2.4.1 Event driven reading with Delphi

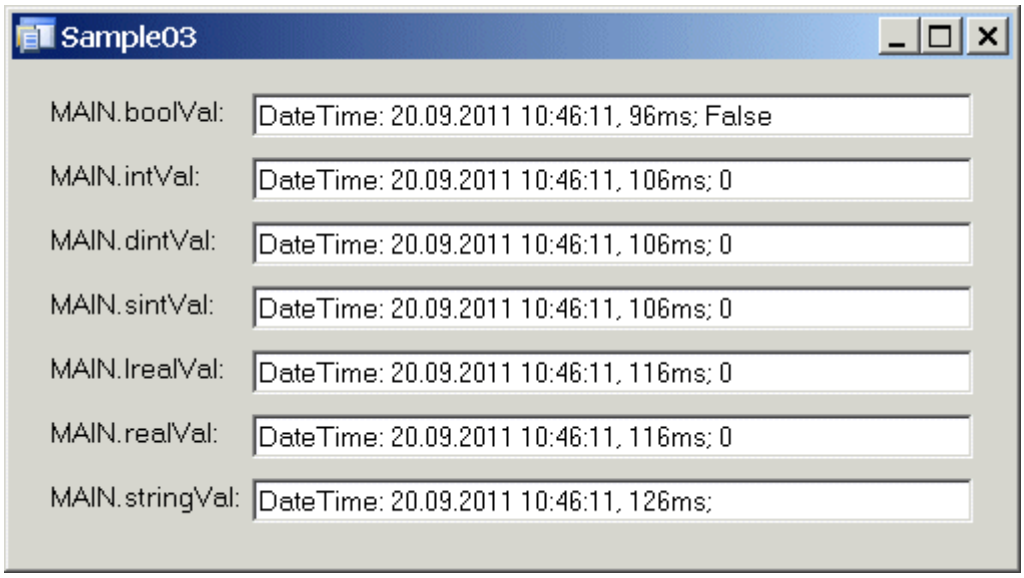
Task

There are 7 global variables in the PLC. Each of these PLC variables is of a different data type. The values of the variables should be read in the most effective manner, and the value with its timestamp is to be displayed on a form.

Description

In the form's *MainForm_Load* event, a connection to each of the PLC variables is created with the TcAdsClient.AddDeviceNotification() [▶ 657] method. The handle for this connection is stored in a array. The parameter TransMode specifies the type of data exchange. AdsTransMode.OnChange has been selected here. This means that the value of the PLC variable is only transmitted if its value within the PLC has changed (see the AdsTransMode [▶ 401] data type). The parameter cycleTime indicates that the PLC is to check whether the corresponding variable has changed every 100 ms. MaxDelay allows to collect notification for a specified interval. If the maxDelay elapse, all notifications will be send at once.
When the PLC variable changes, the TcAdsClient.AdsNotification() [▶ 774] event is called. The parameter e of the event handling method is of the type AdsNotificationEventArgs and contains the time stamp, the handle, the value and the control in which the value is to be displayed. The connections are released again in the *MainForm_FormClosing* event by means of the TcAdsClient.DeleteDeviceNotification() [▶ 697] method. It is essential that you do this, since every connection established by TcAdsClient.AddDeviceNotification() uses resources.
You should also choose appropriate values for the cycle time, since too many write / read operations load the system so heavily that the user interface becomes much slower.

Advice: Don't use time intensive executions in callbacks (OnNotification()).



Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample03;

interface

uses
  System.Drawing,
  System.Collections,
  System.Collections.Generic,
  System.Windows.Forms,
  System.ComponentModel, System.IO, TwinCAT.Ads;

type
  /// <summary>
  /// Summary description for MainForm.
  /// </summary>
```

```
  MainForm = partial class(System.Windows.Forms.Form)
  private
    method tcClient_AdsNotification(sender: Object; e: AdsNotificationEventArgs);
    method MainForm_Load(sender: System.Object; e: System.EventArgs);
    method MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEventArgs)
;
    tcClient    : TcAdsClient;
    hConnect    : Array[0..6] of Integer;
        dataStream  : AdsStream;
        binRead     : BinaryReader;
  protected
    method Dispose(disposing: Boolean); override;
  public
    // boolVal    : BOOL; 1 byte, unsigned integer
    // intVal   : INT; 2 byte, signed integer
    // dintVal  : DINT; 4 byte, signed integer
    // sintVal  : SINT; 1 byte, signed integer
    // lrealVal : LREAL; 8 byte, floating point value
    // realVal  : REAL; 4 byte, floating point value
    // stringVal: STRING(10); 11 byte, string length inclusive terminating zero character
    // Length of the stream = 31 byte, Byte length of all PLC variables
    const MAX_ADSSTREAM_SIZE = 31;

    // max. length of PLC string variable without terminating zero character
    const MAX_STRINGVAL_LENGTH = 10;

    constructor;
  end;

implementation

{$REGION Construction and Disposition}
constructor MainForm;
begin
  //
  // Required for Windows Form Designer support
  //
  InitializeComponent();

  //
  // TODO: Add any constructor code after InitializeComponent call
  //
end;

method MainForm.Dispose(disposing: Boolean);
begin
  if disposing then begin
    if assigned(components) then
      components.Dispose();

    //
    // TODO: Add custom disposition code here
    //
  end;
  inherited Dispose(disposing);
end;
{$ENDREGION}

method MainForm.MainForm_Load(sender: System.Object; e: System.EventArgs);
begin
  try
    // Creaste instance of class TcAdsClient
    tcClient := new TcAdsClient();
    // Create connectin with Port 801 on the local PC
    tcClient.Connect(801);
    // Length of the stream = Byte length of all PLC variables
    dataStream := new AdsStream(MAX_ADSSTREAM_SIZE);
    // Encoding is set to ASCII, to read strings
    binRead := new BinaryReader(dataStream, System.Text.Encoding.ASCII);
    // Add notifications
   hConnect[0] := tcClient.AddDeviceNotification('MAIN.boolVal', dataStream, 0, 1, AdsTransMode.OnCh
ange, 100, 0, tbBool);
        hConnect[1] := tcClient.AddDeviceNotification('MAIN.intVal', dataStream, 1, 2, AdsTransMode.
OnChange, 100, 0, tbInt);
        hConnect[2] := tcClient.AddDeviceNotification('MAIN.dintVal', dataStream, 3, 4, AdsTransMode
.OnChange, 100, 0, tbDint);
        hConnect[3] := tcClient.AddDeviceNotification('MAIN.sintVal', dataStream, 7, 1, AdsTransMode
.OnChange, 100, 0, tbSint);
        hConnect[4] := tcClient.AddDeviceNotification('MAIN.lrealVal', dataStream, 8, 8, AdsTransMod
e.OnChange, 100, 0, tbLreal);
```

```
        hConnect[5] := tcClient.AddDeviceNotification('MAIN.realVal', dataStream, 16, 4, AdsTransMod
e.OnChange, 100, 0, tbReal);
        hConnect[6] := tcClient.AddDeviceNotification('MAIN.stringVal', dataStream, 20, MAX_STRINGVA
L_LENGTH + 1, AdsTransMode.OnChange, 100, 0, tbString);
    // Add notification event handler
    tcClient.AdsNotification += tcClient_AdsNotification
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

method MainForm.MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEvent
Args);
begin
  try
    // Release notifications
    for i : Integer := 0 to 6 do
      tcClient.DeleteDeviceNotification(hConnect[i]);
    // Remove notification handler
    tcClient.AdsNotification -= tcClient_AdsNotification;
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
  // Close connection
  tcClient.Dispose();
end;

method MainForm.tcClient_AdsNotification(sender: Object; e: AdsNotificationEventArgs);
begin
  var strValue : String := '';
  // Convert PLC time stamp to DateTime type
  var time := DateTime.FromFileTime(e.TimeStamp);
  // Get notification data (PLC value) byte offset
  e.DataStream.Position := e.Offset;

  // Convert PLC value to string
  if ( e.NotificationHandle = hConnect[0]) then
    strValue := binRead.ReadBoolean().ToString()
  else if( e.NotificationHandle = hConnect[1] ) then
    strValue := binRead.ReadInt16().ToString()
  else if( e.NotificationHandle = hConnect[2] ) then
    strValue := binRead.ReadInt32().ToString()
  else if( e.NotificationHandle = hConnect[3] ) then
    strValue := binRead.ReadSByte().ToString()
  else if( e.NotificationHandle = hConnect[4] ) then
    strValue := binRead.ReadDouble().ToString()
  else if( e.NotificationHandle = hConnect[5] ) then
    strValue := binRead.ReadSingle().ToString()
  else if ( e.NotificationHandle = hConnect[6] ) then
  begin
    strValue := new String(binRead.ReadChars(MAX_STRINGVAL_LENGTH + 1));
    // Search for terminating zero character and cut at found position
    // Necessary if you want to compare the string to other strings
    strValue := strValue.Substring(0, strValue.IndexOf(Char(0)));
  end
  else
    ;


  TextBox(e.UserData).Text := System.String.Format('DateTime: {0}, {1}
ms; {2}', [time, time.Millisecond, strValue]);
end;

end.
```

PLC program

```
PROGRAM MAIN
VAR
    boolVal : BOOL;
    intVal : INT;
    dintVal : DINT;
    sintVal : SINT;
    lrealVal : LREAL;
```

```
    realVal : REAL;
    stringVal : STRING(10);
END_VAR
```

```
;
```

Requirements

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466760331/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466761739/.exe

6.2.4.2 Event driven reading with Visual Basic

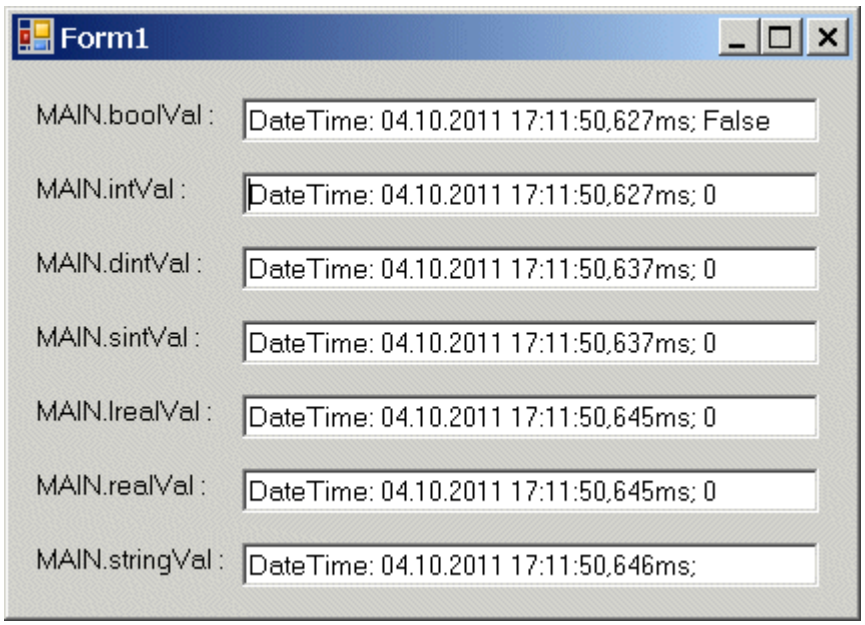
Task

There are 7 global variables in the PLC. Each of these PLC variables is of a different data type. The values of the variables should be read in the most effective manner, and the value with its timestamp is to be displayed on a form.

Description

In the form's *MainForm_Load* event, a connection to each of the PLC variables is created with the TcAdsClient.AddDeviceNotification() [▶ 657] method. The handle for this connection is stored in a array. The parameter TransMode specifies the type of data exchange. AdsTransMode.OnChange has been selected here. This means that the value of the PLC variable is only transmitted if its value within the PLC has changed (see the AdsTransMode [▶ 401] data type). The parameter cycleTime indicates that the PLC is to check whether the corresponding variable has changed every 100 ms. MaxDelay allows to collect notification for a specified interval. If the maxDelay elapse, all notifications will be send at once.
When the PLC variable changes, the TcAdsClient.AdsNotification() [▶ 774] event is called. The parameter e of the event handling method is of the type AdsNotificationEventArgs and contains the time stamp, the handle, the value and the control in which the value is to be displayed. The connections are released again in the *MainForm_FormClosing* event by means of the TcAdsClient.DeleteDeviceNotification() [▶ 697] method. It is essential that you do this, since every connection established by TcAdsClient.AddDeviceNotification() uses resources.
You should also choose appropriate values for the cycle time, since too many write / read operations load the system so heavily that the user interface becomes much slower.

Advice: Don't use time intensive executions in callbacks (OnNotification()).



Visual Basic (for .NET framework) program

```
Imports TwinCAT.Ads
Imports System.IO

Public Class Form1
    Inherits System.Windows.Forms.Form

#Region " Windows Form Designer generated code "
...

#End Region

    ' Handle events (1): without withevents declaration
    Private tcClient As TcAdsClient
    ' Handle events (2):You can also declare tcClient as:
    ' Private WithEvents tcClient As TcAdsClient
    Private hConnect() As Integer
    Private dataStream As AdsStream
    Private binRead As BinaryReader

    Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles MyBase.Load
    Dim dataStream As New AdsStream(31)
    ' Set encoding to ASCII to be able to read strings
    binRead = New BinaryReader(dataStream, System.Text.Encoding.ASCII)
    ' Create instance of TcAdsClient class
    tcClient = New TcAdsClient()
    ' Establish connection to local port 801
    tcClient.Connect(801)

    ReDim hConnect(7)

    Try
        hConnect(0) = tcClient.AddDeviceNotification("MAIN.boolVal", dataStream, 0, 1, _
        AdsTransMode.OnChange, 100, 0, tbBool)
        hConnect(1) = tcClient.AddDeviceNotification("MAIN.intVal", dataStream, 1, 2, _
        AdsTransMode.OnChange, 100, 0, tbInt)
        hConnect(2) = tcClient.AddDeviceNotification("MAIN.dintVal", dataStream, 3, 4, _
        AdsTransMode.OnChange, 100, 0, tbDint)
        hConnect(3) = tcClient.AddDeviceNotification("MAIN.sintVal", dataStream, 7, 1, _
        AdsTransMode.OnChange, 100, 0, tbSint)
        hConnect(4) = tcClient.AddDeviceNotification("MAIN.lrealVal", dataStream, 8, 8, _
        AdsTransMode.OnChange, 100, 0, tbLreal)
        hConnect(5) = tcClient.AddDeviceNotification("MAIN.realVal", dataStream, 16, 4, _
        AdsTransMode.OnChange, 100, 0, tbReal)
        hConnect(6) = tcClient.AddDeviceNotification("MAIN.stringVal", dataStream, 20, 11, _
        AdsTransMode.OnChange, 100, 0, tbString)

        ' Handle events (1):
        AddHandler tcClient.AdsNotification, AddressOf OnNotification
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
```

```
    End Sub

    Private Sub Form1_Closing(ByVal sender As Object, ByVal e As System.ComponentModel.CancelEventAr
gs) Handles MyBase.Closing
    Try
        tcClient.Dispose()
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub

    ' Handle events (2):
    ' Private Sub OnNotification(ByVal sender As Object, ByVal e As AdsSyncNotificationEventArgs) Ha
ndles tcClient.AdsSyncNotification
    ' Handle events (1):
    Private Sub OnNotification(ByVal sender As Object, ByVal e As AdsNotificationEventArgs)
    Dim time As New DateTime()
    time = DateTime.FromFileTime(e.TimeStamp)
    e.DataStream.Position = e.Offset
    Dim strValue As String = ""

    If (e.NotificationHandle = hConnect(0)) Then
        strValue = binRead.ReadBoolean().ToString()
    ElseIf (e.NotificationHandle = hConnect(1)) Then
        strValue = binRead.ReadInt16().ToString()
    ElseIf (e.NotificationHandle = hConnect(2)) Then
        strValue = binRead.ReadInt32().ToString()
    ElseIf (e.NotificationHandle = hConnect(3)) Then
        strValue = binRead.ReadByte().ToString()
    ElseIf (e.NotificationHandle = hConnect(4)) Then
        strValue = binRead.ReadDouble().ToString()
    ElseIf (e.NotificationHandle = hConnect(5)) Then
        strValue = binRead.ReadSingle().ToString()
    ElseIf (e.NotificationHandle = hConnect(6)) Then
        strValue = New String(binRead.ReadChars(11))
    End If

    Dim tb As New TextBox()
    tb = CType(e.UserData, TextBox)
    tb.Text = String.Format("DateTime: {0},{1}ms; {2}", time, time.Millisecond, strValue)
    End Sub
End Class
```

PLC program

```
PROGRAM MAIN
VAR
    boolVal : BOOL;
    intVal : INT;
    dintVal : DINT;
    sintVal : SINT;
    lrealVal : LREAL;
    realVal : REAL;
    stringVal : STRING(10);
END_VAR
```

```
;
```

Requirements

Language / IDE	Unpack the sample program
Visual Basic (for .NET framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12463812107/.exe

6.2.5 Reading and writing of string variables

Download

Language / IDE	Unpack the sample program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490130187/.zip
Visual Basic (for .NET Framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490131595/.exe **TwinCAT.Ads.NET v1.0.0.10 and higher**: https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490133003/.exe
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490134411/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490135819/.exe

Task

A .Net application should read a string from the PLC and write a string to the PLC.

Description

The PLC contains the string MAIN.text.

In the Form1_Load event method a new instance of the class TcAdsClient is created. Then the method TcAdsClient.Connect [▶ 689] of the TcAdsClient object is called to establish a connection to the port 801. Finally the method TcAdsClient.CreateVariableHandle [▶ 696] is used to fetch the handle of the PLC variable. When the program finishes, the Dispose method of the TcAdsClient object is called.

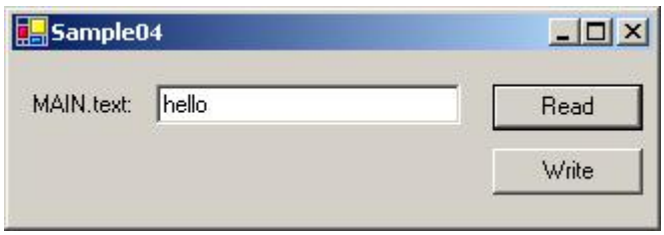
When the user clicks the "Read" button on the form, the string is read by means of the TcAdsClient.Read [▶ 708] method and is displayed in the text box. When the user clicks the "Write" button on the form, the string is wirtten to PLC and is displayed in the text box.

TwinCAT.Ads.NET version < 1.0.0.10:

The length of the AdsStream must be equal to size of the string in the PLC plus 1 (inclusive the terminating zero character). The parameter System.Text.Encoding.ASCII has to be passed to constructor of the BinaryReader class. This is necessary for the BinaryReader to be able to convert the ANSI characters to the char type.

TwinCAT.Ads.NET version >= 1.0.0.10:

The classes AdsBinaryReader [▶ 112] and AdsBinaryWriter [▶ 121] can be used to read and write strings (see commented section in sample program). These classes derive from the BinaryReader/Writer classes. To read a string from the stream one must call the method AdsBinaryReader.ReadPlcString [▶ 120]. To write a string to the stream one must call the method AdsBinaryWriter.WritePLCString [▶ 131]. The length of the AdsStream must be equal to size of the string in the PLC (without terminating zero character). The length is passed to this method and have to be equal to the length of the string in the PLC (without the terminating zero character).



C# program

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using TwinCAT.Ads;
using System.IO;

namespace Sample04
{
    public class Form1 : System.Windows.Forms.Form
    {
    private System.ComponentModel.Container components = null;
    private System.Windows.Forms.TextBox textBox1;
    private System.Windows.Forms.Button btnRead;
    private System.Windows.Forms.Button btnWrite;
    private System.Windows.Forms.Label label1;
    private TcAdsClient adsClient;
    private int varHandle;

    public Form1()
    {
       InitializeComponent();
    }

    protected override void Dispose( bool disposing )
    {
    ...
    }

    private void InitializeComponent()
    {
    ...
    }

    [STAThread]
    static void Main()
    {
        Application.Run(new Form1());
    }

        private void Form1_Load(object sender, System.EventArgs e)
    {
        try
        {
        adsClient = new TcAdsClient();

           // PLC1 Port: TwinCAT2=801, TwinCAT3=851
           tcClient.Connect(801);

        varHandle = adsClient.CreateVariableHandle("MAIN.text");
        }
        catch( Exception err)
        {
        MessageBox.Show(err.Message);
        }
    }

    private void Form1_Closing(object sender, System.ComponentModel.CancelEventArgs e)
    {
        adsClient.Dispose();
    }

    private void btnRead_Click(object sender, System.EventArgs e)
    {
        try
```

```
        {
        //length of the stream = length of string in sps + 1
        AdsStream adsStream = new AdsStream(31);
        BinaryReader reader = new BinaryReader(adsStream,System.Text.Encoding.ASCII);

        int length = adsClient.Read(varHandle, adsStream);
        string text = new string(reader.ReadChars(length));
        //necessary if you want to compare the string to other strings
        //text = text.Substring(0,text.IndexOf('\0'));
        textBox1.Text = text;
       }
        catch(Exception err)
        {
        MessageBox.Show(err.Message)
        }
    }

    private void btnWrite_Click(object sender, System.EventArgs e)
    {
        try
        {
        //length of the stream = length of string + 1
        AdsStream adsStream = new AdsStream(textBox1.Text.Length+1);
        BinaryWriter writer = new BinaryWriter(adsStream, System.Text.Encoding.ASCII);
        writer.Write(textBox1.Text.ToCharArray());
        //add terminating zero
        writer.Write('\0');
        adsClient.Write(varHandle,adsStream);
        }
        catch(Exception err)
        {
        MessageBox.Show(err.Message);
        }
    }


/*From version 1.0.0.10 and higher the classes AdsBinaryReader and AdsBinaryWriter
  can be used to read and write strings
        private void btnRead_Click(object sender, System.EventArgs e)
     {
        try
        {
        AdsStream adsStream = new AdsStream(30);
        AdsBinaryReader reader = new AdsBinaryReader(adsStream);
        adsClient.Read(varHandle, adsStream);
        textBox1.Text = reader.ReadPlcString(30);
        }
        catch(Exception err)
        {
        MessageBox.Show(err.Message);
        }
     }

    private void btnWrite_Click(object sender, System.EventArgs e)
    {
        try
        {
        AdsStream adsStream = new AdsStream(30);
        AdsBinaryWriter writer = new AdsBinaryWriter(adsStream);
        writer.WritePlcString(textBox1.Text, 30);
        adsClient.Write(varHandle, adsStream);
        }
        catch(Exception err)
        {
        MessageBox.Show(err.Message);
        }
    }
*/

    }
}
```

PLC program

```
PROGRAM MAIN
VAR
 text : STRING[30] := 'hello';
END_VAR
```

6.2.5.1 Reading and writing of string variables with Delphi

Task

A .Net application should read a string from the PLC and write a string to the PLC.

Description

The PLC contains the string: MAIN.text.

In the *MainForm_Load* event method a new instance of the class TcAdsClient is created. Then the method TcAdsClient.Connect [▶ 689] of the TcAdsClient object is called to establish a connection to the port 801. Finally the method TcAdsClient.CreateVariableHandle [▶ 696] is used to fetch the handle of the PLC variable . When the program finishes, the Dispose method of the TcAdsClient object is called. When the user clicks the "Read" button on the form, the string is read by means of the TcAdsClient.Read [▶ 708] method and is displayed in the text box. When the user clicks the "Write" button ont the form, the string is wirtten to PLC and is displayed in the text box.

TwinCAT.Ads.NET version < 1.0.0.10:

The length of the AdsStream must be equal to size of the string in the PLC plus 1 (inclusive the terminating zero character). The parameter System.Text.Encoding.ASCII has to be passed to constructor of the BinaryReader class. This is necessary for the BinaryReader to be able to convert the ANSI characters to the char type.

TwinCAT.Ads.NET version >= 1.0.0.10:

The classes AdsBinaryReader [▶ 112] and AdsBinaryWriter [▶ 121] can be used to read and write strings (see commented section in sample program). These classes derive from the BinaryReader/Writer classes. To read a string from the stream one has to call the method AdsBinaryReader.ReadPlcString. To write a string to the stream one has to call the method AdsBinaryWriter.WritePLCString. The length of the AdsStream must be equal to size of the string in the PLC (without terminating zero character). The length is passed to this method and must be equal to the length of the string in the PLC (without the terminating zero character).

Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample04;

interface

uses
```

```
  System.Drawing,
  System.Collections,
  System.Collections.Generic,
  System.Windows.Forms,
  System.ComponentModel, System.IO, TwinCAT.Ads, System.Text;//////////////////////////////////////
//////////////////////////////////
// Define OLD_ADSNET_INTF only if you are using Ads.NET version < 1.0.0.10
// From Ads.NET version >= 1.0.0.10 and higher the classes
// AdsBinaryReader and AdsBinaryWriter can be used to read and write strings
// {$DEFINE OLD_ADSNET_INTF}
////////////////////////////////////////////////////////////////////////

type
  /// <summary>
  /// Summary description for MainForm.
  /// </summary>
  MainForm = partial class(System.Windows.Forms.Form)
  private
    adsClient   : TcAdsClient;
    varHandle   : Integer;
    method MainForm_Load(sender: System.Object; e: System.EventArgs);
    method MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEventArgs)
;
    method btnRead_Click(sender: System.Object; e: System.EventArgs);
    method btnWrite_Click(sender: System.Object; e: System.EventArgs);
  protected
    method Dispose(disposing: Boolean); override;
  public
    // text : STRING[30] := 'hello';
    // Byte length of the PLC string (without terminating zero character)
    const MAX_TEXT_LENGTH  = 30;
    constructor;
  end;

implementation

{$REGION Construction and Disposition}
constructor MainForm;
begin
  //
  // Required for Windows Form Designer support
  //
  InitializeComponent();

  //
  // TODO: Add any constructor code after InitializeComponent call
  //
end;

method MainForm.Dispose(disposing: Boolean);
begin
  if disposing then begin
    if assigned(components) then
      components.Dispose();

    //
    // TODO: Add custom disposition code here
    //
  end;
  inherited Dispose(disposing);
end;
{$ENDREGION}

method MainForm.MainForm_Load(sender: System.Object; e: System.EventArgs);
begin
  try
    // Create instance of class TcAdsClient
    adsClient := new TcAdsClient();
    // Create connection with Port 801 on local PC
    adsClient.Connect(801);
    // Create PLC variable handle
    varHandle := adsClient.CreateVariableHandle('MAIN.text');
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source);
  end;
end;

method MainForm.MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEvent
Args);
```

```
begin
  try
    // Release PLC variable handle
    adsClient.DeleteVariableHandle(varHandle);
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
  // Close connection
  adsClient.Dispose();
end;

{$IFNDEF OLD_ADSNET_INTF} // Ads.NET version >= 1.0.0.10 (new)
method MainForm.btnRead_Click(sender: System.Object; e: System.EventArgs);
begin
  try
    // Create stream to hold the data. Length of the stream = Length of string in PLC
    var dataStream := new AdsStream(MAX_TEXT_LENGTH);
    // Create AdsBinaryReader. Encoding is set to ASCII, to read strings
    var reader := new AdsBinaryReader(dataStream);
    // Read string characters from PLC to the stream
    adsClient.Read(varHandle, dataStream);
    // Read stream characters from stream
    textBox1.Text := reader.ReadPlcString(MAX_TEXT_LENGTH)
  except
    on err : Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
 end;

method MainForm.btnWrite_Click(sender: System.Object; e: System.EventArgs);
begin
  try
    // Create stream to hold the data. Length of the stream = Length of string in PLC
    var dataStream := new AdsStream(MAX_TEXT_LENGTH);
    // Create AdsBinaryWriter. Encoding is set to ASCII, to read strings
    var writer := new AdsBinaryWriter(dataStream);
    // Add string characters to the stream
    writer.WritePlcString(textBox1.Text, MAX_TEXT_LENGTH);
    // Write stream to the PLC
    adsClient.Write(varHandle, dataStream)
  except
    on err : Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;


{$ELSE} // OLD_ADSNET_INTF defined => Ads.NET version < 1.0.0.10
method MainForm.btnRead_Click(sender: System.Object; e: System.EventArgs);
begin
  try
    // Length of the stream = Length of string in PLC + 1
    var dataStream := new AdsStream(MAX_TEXT_LENGTH + 1);
    // Create BinaryReader. Encoding is set to ASCII, to read strings
    var reader := new BinaryReader(dataStream, System.Text.Encoding.ASCII);
    // Read string characters from PLC to the stream
    var length := adsClient.Read(varHandle, dataStream);
    // Read stream characters from stream
    var text := new String(reader.ReadChars(length));
    // Search for terminating zero character and cut at found position
    // Necessary if you want to compare the string to other strings
    text := text.Substring(0,text.IndexOf(Char(0)));
    textBox1.Text := text
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

method MainForm.btnWrite_Click(sender: System.Object; e: System.EventArgs);
begin
  try
    // Length of the stream = Length of string in PLC + 1
    var dataStream := new AdsStream(MAX_TEXT_LENGTH + 1);
    // Create BinaryWriter. Encoding is set to ASCII, to read strings
    var writer := new BinaryWriter(dataStream, System.Text.Encoding.ASCII);
    // Add string characters to the stream
    writer.Write(textBox1.Text.ToCharArray());
    // Add terminating zero to the stream
```

```
    writer.Write(Char(0));
    // Write stream to the PLC
    adsClient.Write(varHandle, dataStream)
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;
{$ENDIF}


end.
```

PLC program

```
PROGRAM MAIN
VAR
    text : STRING[30] := 'hello';
END_VAR
```

```
;
```

Download

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466763147/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466764555/.exe

6.2.5.2 Reading and writing of string variables with Visual Basic

Task

A .Net application should read a string from the PLC and write a string to the PLC.

Description

The PLC contains the string: MAIN.text.

In the *MainForm_Load* event method a new instance of the class TcAdsClient is created. Then the method TcAdsClient.Connect [▶ 689] of the TcAdsClient object is called to establish a connection to the port 801. Finally the method TcAdsClient.CreateVariableHandle [▶ 696] is used to fetch the handle of the PLC variable . When the program finishes, the Dispose method of the TcAdsClient object is called. When the user clicks the "Read" button on the form, the string is read by means of the TcAdsClient.Read [▶ 708] method and is displayed in the text box. When the user clicks the "Write" button ont the form, the string is wirtten to PLC and is displayed in the text box.

TwinCAT.Ads.NET version < 1.0.0.10:

The length of the AdsStream must be equal to size of the string in the PLC plus 1 (inclusive the terminating zero character). The parameter System.Text.Encoding.ASCII has to be passed to constructor of the BinaryReader class. This is necessary for the BinaryReader to be able to convert the ANSI characters to the char type.

TwinCAT.Ads.NET version >= 1.0.0.10:

The classes AdsBinaryReader [▶ 112] and AdsBinaryWriter [▶ 121] can be used to read and write strings (see commented section in sample program). These classes derive from the BinaryReader/Writer classes. To read a string from the stream one has to call the method AdsBinaryReader.ReadPlcString. To write a string to the stream one has to call the method AdsBinaryWriter.WritePLCString. The length of the AdsStream must be equal to size of the string in the PLC (without terminating zero character). The length is passed to these method and have to be equal to the length of the string in the PLC (without the terminating zero character).

Visual Basic (for .NET framework) program

```
Imports TwinCAT.Ads
Imports System.IO

Public Class Form1
    Inherits System.Windows.Forms.Form

    Private adsClient As TcAdsClient
    Private varHandle As Integer

#Region " Windows Form Designer generated code "

...

#End Region
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBas
e.Load
    Try
        adsClient = New TcAdsClient
        adsClient.Connect(801)
        varHandle = adsClient.CreateVariableHandle("MAIN.text")
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub

    Private Sub Form1_Closing(ByVal sender As Object, ByVal e As System.ComponentModel.CancelEventAr
gs) Handles MyBase.Closing
    adsClient.Dispose()
    End Sub


    Private Sub btnRead_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles bt
nRead.Click
    Dim length As Integer
    Dim text As String
    Dim dataStream As AdsStream
    Dim reader As BinaryReader
    Try
        'length of the stream = length of string in sps + 1
        dataStream = New AdsStream(31)
        reader = New BinaryReader(dataStream, System.Text.Encoding.ASCII)
        length = adsClient.Read(varHandle, dataStream)
        text = New String(reader.ReadChars(length))
        'necessary if you want to compare the string to other strings
        'text = text.Substring(0, text.IndexOf(Chr(0)))
        textBox1.Text = text
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub
```

```
    Private Sub btnWrite_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles b
tnWrite.Click
    Dim dataStream As AdsStream
    Dim writer As BinaryWriter
    Dim c As Char
    Try
        'length of the stream = length of string + 1
        dataStream = New AdsStream(textBox1.Text.Length + 1)
        writer = New BinaryWriter(dataStream, System.Text.Encoding.ASCII)
        writer.Write(textBox1.Text.ToCharArray())
        'add terminating zero
        writer.Write(Chr(0))
        adsClient.Write(varHandle, dataStream)
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try

    End Sub

End Class
```

PLC program

```
PROGRAM MAIN
VAR
    text : STRING[30] := 'hello';
END_VAR
```

```
;
```

Requirements

Language / IDE	Unpack the sample program
Visual Basic (for .NET framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12463813515/.exe **TwinCAT.Ads.NET v1.0.0.10 and higher** https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12463814923/.exe

6.2.6 Reading and writing of TIME/DATE variables

Download

Language / IDE	Unpack the example program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490137227/.zip
Visual Basic (for .NET Framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490138635/.exe
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490140043/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490141451/.exe

Task

A .Net application should read and write a date and a time.

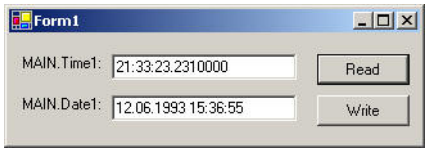
Description

The PLC contains the TIME variable MAIN.Time1 and the DT variable MAIN.Date1.

In the Form1_Load event method a new instance of the class TcAdsClient is created. Then the method TcAdsClient.Connect [▶ 689] of the TcAdsClient object is called to establish a connection to the port 801. Finally the method TcAdsClient.CreateVariableHandle [▶ 696] is used to fetch the handle of the PLC variables. When the program finishes, the Dispose method of the TcAdsClient object is called.

When the user clicks the "Read" button on the form, the variables MAIN.Time1 and MAIN.Date1 are read by means of the TcAdsClient.Read [▶ 708] method. The class AdsBinaryReader [▶ 112], that derives from BinaryReader, is used to read the time and date from the PLC. The method AdsBinaryReader.ReadPlcTIME [▶ 120] reads the time from the AdsStream and converts it to the .NET type TimeSpan. The method AdsBinaryReader.ReadPlcDATE [▶ 119] reads the date from the AdsStream and converts it to the .NET type DateTime.

When the user clicks the "Write" button ont the form, the variables MAIN.Time1 and MAIN.Date1 are wirtten to the PLC. The class AdsBinaryWriter [▶ 121], that derives from BinaryWriter, is used to write the time and date to the PLC. The method AdsBinaryWriter.WritePlcType [▶ 130] writes the .NET types TimeSpan and DateTime in the PLC format for time and date types to the AdsStream.



C# program

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using TwinCAT.Ads;
using System.IO;

namespace Sample04
{
    public class Form1 : System.Windows.Forms.Form
    {
    private System.ComponentModel.Container components = null;
    private System.Windows.Forms.TextBox textBox1;
    private System.Windows.Forms.Button btnRead;
    private System.Windows.Forms.Button btnWrite;
    private System.Windows.Forms.Label label1;
    private TcAdsClient adsClient;
    private int[] varHandles;

    public Form1()
    {
       InitializeComponent();
    }

    protected override void Dispose( bool disposing )
    {
    ...
    }

    private void InitializeComponent()
    {
    ...
    }

    [STAThread]
    static void Main()
    {
        Application.Run(new Form1());
    }

        private void Form1_Load(object sender, System.EventArgs e)
    {
        try
        {
```

```
        adsClient = new TcAdsClient();

        // PLC1 Port: TwinCAT2=801, TwinCAT3=851
        adsClient.Connect(801);
        varHandles = new int[2];
        varHandles[0] = adsClient.CreateVariableHandle("MAIN.Time1");
        varHandles[1] = adsClient.CreateVariableHandle("MAIN.Date1");
        }
        catch( Exception err)
        {
        MessageBox.Show(err.Message);
        }
    }

    private void Form1_Closing(object sender, System.ComponentModel.CancelEventArgs e)
    {
        adsClient.Dispose();
    }

    private void btnRead_Click(object sender, System.EventArgs e)
    {
        try
        {
        AdsStream adsStream = new AdsStream(4);
        AdsBinaryReader reader = new AdsBinaryReader(adsStream);
        adsClient.Read(varHandles[0], adsStream);
        textBox1.Text = reader.ReadPlcTIME().ToString();
        adsStream.Position = 0;
        adsClient.Read(varHandles[1], adsStream);
        textBox2.Text = reader.ReadPlcDATE().ToString();
       }
        catch(Exception err)
        {
        MessageBox.Show(err.Message)
        }
    }

    private void btnWrite_Click(object sender, System.EventArgs e)
    {
        try
        {
        AdsStream adsStream = new AdsStream(4);
        AdsBinaryWriter writer = new AdsBinaryWriter(adsStream);
        writer.WritePlcType(TimeSpan.Parse(textBox1.Text));
        adsClient.Write(varHandles[0], adsStream);

        adsStream.Position = 0;
        writer.WritePlcType(DateTime.Parse(textBox2.Text));
        adsClient.Write(varHandles[1], adsStream)
        }
        catch(Exception err)
        {
        MessageBox.Show(err.Message);
        }
    }
    }
}
```

PLC program

```
PROGRAM MAIN
VAR
    Time1:TIME := T#21h33m23s231ms;
    Date1:DT:=DT#1993-06-12-15:36:55.40;
END_VAR
```

6.2.6.1 Reading and writing of DATE/TIME variables with Delphi

From TwinCAT.Ads.NET version >= 1.0.0.10

Task

A .Net application should read and write a date and a time.

Description

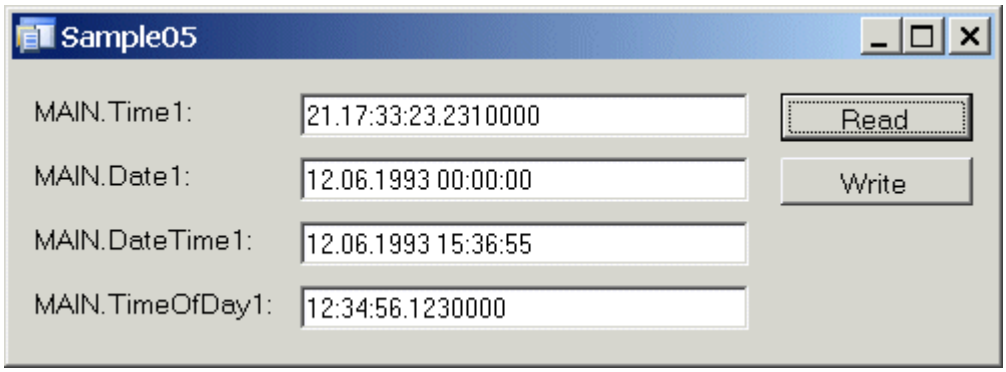
The PLC program contains variables of following type:

- MAIN.Time1 vom Typ TIME;
- MAIN.Date1 vom Typ DATE;
- MAIN.DateTime1 vom Typ DT (DATE_AND_TIME);
- MAIN.TimeOfDay1 vom Typ TOD (TIME_OF_DAY);

In the *MainForm_Load* event method a new instance of the class TcAdsClient is created. Then the method TcAdsClient.Connect [▶ 689] of the TcAdsClient object is called to establish a connection to the port 801. Finally the method TcAdsClient.CreateVariableHandle [▶ 696] is used to fetch the handle of the PLC variables . When the program finishes, the *MainForm_FormClosing* event is called and the Dispose method of the TcAdsClient object is called.

When the user clicks the "Read" button on the form, the variables MAIN.Time1, MAIN.Date1 and so on are read by means of the TcAdsClient.Read [▶ 708] method. The class AdsBinaryReader [▶ 112], that derives from BinaryReader, is used to read the time and date from the PLC. The method AdsBinaryReader.ReadPlcTIME [▶ 120] reads the time from the AdsStream and converts it to the .NET type: *TimeSpan*. The method AdsBinaryReader.ReadPlcDATE [▶ 119] reads the date from the AdsStream and converts it to the .NET type: *DateTime*.

When the user clicks the "Write" button ont the form, the variables MAIN.Time1 and MAIN.Date1 are wirtten to the PLC. The class AdsBinaryWriter [▶ 121], that derives from BinaryWriter, is used to write the time and date to the PLC. The method AdsBinaryWriter.WritePlcType [▶ 130] writes the .NET types TimeSpan and DateTime in the PLC format for time and date types to the AdsStream.



Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample05;

interface

uses
  System.Drawing,
  System.Collections,
  System.Collections.Generic,
  System.Windows.Forms,
  System.ComponentModel, System.IO, TwinCAT.Ads;

type
  /// <summary>
  /// Summary description for MainForm.
  /// </summary>
  MainForm = partial class(System.Windows.Forms.Form)
  private
    adsClient   : TcAdsClient;
    varHandles  : Array Of Integer;
    method MainForm_Load(sender: System.Object; e: System.EventArgs);
    method MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEventArgs)
;
```

```
    method btnRead_Click(sender: System.Object; e: System.EventArgs);
    method btnWrite_Click(sender: System.Object; e: System.EventArgs);
  protected
    method Dispose(disposing: Boolean); override;
  public
    // Time1    :TIME; Time duration, 4 byte unsigned integer, resolution in milliseconds
    // Date1      :DATE; Date, 4 byte unsigned integer, resolution in seconds since 01.01.1970
    // DateTime1      :DATE_AND_TIME; Date and time, 4 byte unsigned integer, resolution in seconds
since 01.01.1970
    // TimeOfDay1     :TIME_OF_DAY; Time of day, 4 byte unsigned integer, resolution in milliseconds
    const MAX_ADSSTREAM_SIZE = 4;// SIZEOF(Time1) = SIZEOF( Date1) = SIZEOF(DateTime1) = SIZEOF(Time
OfDay1)
    constructor;
  end;

implementation

{$REGION Construction and Disposition}
constructor MainForm;
begin
  //
  // Required for Windows Form Designer support
  //
  InitializeComponent();

  //
  // TODO: Add any constructor code after InitializeComponent call
  //
end;

method MainForm.Dispose(disposing: Boolean);
begin
  if disposing then begin
    if assigned(components) then
      components.Dispose();

    //
    // TODO: Add custom disposition code here
    //
  end;
  inherited Dispose(disposing);
end;
{$ENDREGION}

method MainForm.MainForm_Load(sender: System.Object; e: System.EventArgs);
begin
  try
    // Set some initialization values
    TextBox1.Text := new TimeSpan(0, 10, 10, 12, 345).ToString();
    TextBox2.Text := new DateTime(2011, 9, 14).ToString();
    TextBox3.Text := new DateTime(2011, 9, 14, 11, 6, 21).ToString();
    TextBox4.Text := new TimeSpan(10, 11, 22).ToString();
    // Creaste instance of class TcAdsClient
    adsClient := new TcAdsClient();
    // Create connectin with Port 801 on the local PC
    adsClient.Connect(801);
    varHandles := new Integer[4];
    // Create PLC variable handles
    varHandles[0] := adsClient.CreateVariableHandle('MAIN.Time1');
    varHandles[1] := adsClient.CreateVariableHandle('MAIN.Date1');
    varHandles[2] := adsClient.CreateVariableHandle('MAIN.DateTime1');
    varHandles[3] := adsClient.CreateVariableHandle('MAIN.TimeOfDay1')
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

method MainForm.MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEvent
Args);
begin
  try
    // Release PLC variable handles
    for i:Integer := varHandles.GetLowerBound(0) to varHandles.GetUpperBound(0) do
      adsClient.DeleteVariableHandle(varHandles[i]);
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
  // Close connection
```

```
  adsClient.Dispose();
end;

method MainForm.btnRead_Click(sender: System.Object; e: System.EventArgs);
begin
  try
    // Create data stream instance
    var dataStream := new AdsStream(MAX_ADSSTREAM_SIZE);
    // Create AdsBinaryReader instance
    var binRead := new AdsBinaryReader(dataStream);
    // Read TIME value
    dataStream.Position := 0;
    adsClient.Read(varHandles[0], dataStream);
    textBox1.Text := binRead.ReadPlcTIME().ToString();
    // Read DATE value
    dataStream.Position := 0;
    adsClient.Read(varHandles[1], dataStream);
    textBox2.Text := binRead.ReadPlcDATE().Date.ToString();
    // Read DATE_AND_TIME value
    dataStream.Position := 0;
    adsClient.Read(varHandles[2], dataStream);
    textBox3.Text := binRead.ReadPlcDATE().ToString();
    // Read TIME_OF_DAY value
    dataStream.Position := 0;
    adsClient.Read(varHandles[3], dataStream);
    textBox4.Text := binRead.ReadPlcTime().ToString()
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

method MainForm.btnWrite_Click(sender: System.Object; e: System.EventArgs);
begin
  try
    // Create data stream instance
    var dataStream := new AdsStream(MAX_ADSSTREAM_SIZE);
    // Create AdsBinaryWriter instance
    var binWrite := new AdsBinaryWriter(dataStream);
    // Write TIME value
    dataStream.Position := 0;
    binWrite.WritePlcType(TimeSpan.Parse(textBox1.Text));
    adsClient.Write(varHandles[0], dataStream);
    // Write DATE value
    dataStream.Position := 0;
    binWrite.WritePlcType(DateTime.Parse(textBox2.Text));
    adsClient.Write(varHandles[1], dataStream);
    // Write DATE_AND_TIME value
    dataStream.Position := 0;
    binWrite.WritePlcType(DateTime.Parse(textBox3.Text));
    adsClient.Write(varHandles[2], dataStream);
    // Write TIME_OF_DATE value
    dataStream.Position := 0;
    binWrite.WritePlcType(TimeSpan.Parse(textBox4.Text));
    adsClient.Write(varHandles[3], dataStream)
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

end.
```

PLC program

```
PROGRAM MAIN
VAR
    Time1   :TIME := T#521h33m23s231ms;
    Date1   :DATE := D#1993-06-12;
    DateTime1 :DATE_AND_TIME := DT#1993-06-12-15:36:55;
    TimeOfDay1 :TIME_OF_DAY := TOD#12:34:56.123;
END_VAR
```

```
;
```

Download

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466765963/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466767371/.exe

6.2.6.2 Reading and writing of DATE/TIME variables with Visual Basic

From TwinCAT.Ads.NET version >= 1.0.0.10

Task

A .Net application should read and write a date and a time.

Description

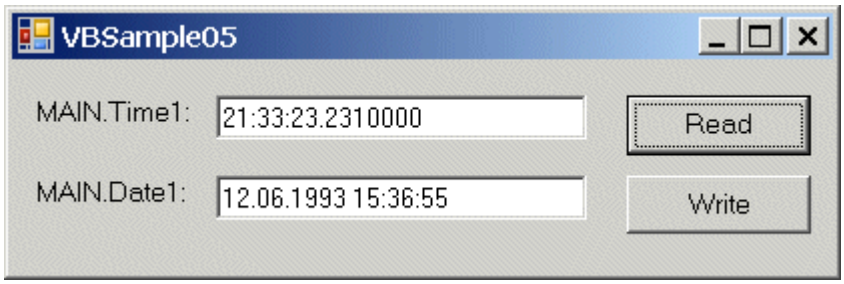
The PLC program contains variables of following type:

- MAIN.Time1 vom Typ TIME;
- MAIN.Date1 vom Typ DATE;

In the *MainForm_Load* event method a new instance of the class TcAdsClient is created. Then the method TcAdsClient.Connect [▶ 689] of the TcAdsClient object is called to establish a connection to the port 801. Finally the method TcAdsClient.CreateVariableHandle [▶ 696] is used to fetch the handle of the PLC variables . When the program finishes, the *MainForm_FormClosing* event is called and the Dispose method of the TcAdsClient object is called.

When the user clicks the "Read" button on the form, the variables MAIN.Time1, MAIN.Date1 and so on are read by means of the TcAdsClient.Read [▶ 708] method. The class AdsBinaryReader [▶ 112], that derives from BinaryReader, is used to read the time and date from the PLC. The method AdsBinaryReader.ReadPlcTIME [▶ 120] reads the time from the AdsStream and converts it to the .NET type: *TimeSpan*. The method AdsBinaryReader.ReadPlcDATE [▶ 119] reads the date from the AdsStream and converts it to the .NET type: *DateTime*.

When the user clicks the "Write" button ont the form, the variables MAIN.Time1 and MAIN.Date1 are wirtten to the PLC. The class AdsBinaryWriter [▶ 121], that derives from BinaryWriter, is used to write the time and date to the PLC. The method AdsBinaryWriter.WritePlcType [▶ 130] writes the .NET types TimeSpan and DateTime in the PLC format for time and date types to the AdsStream.



Visual Basic (for .NET framework) program

```
Imports TwinCAT.Ads
Imports System.IO

Public Class Form1
    Inherits System.Windows.Forms.Form
```

```
    Private adsClient As TcAdsClient
    Private varHandles(1) As Integer

#Region " Windows Form Designer generated code "

...

#End Region
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBas
e.Load
    Try
        adsClient = New TcAdsClient
        adsClient.Connect(801)
        varHandles(0) = adsClient.CreateVariableHandle("MAIN.Time1")
        varHandles(1) = adsClient.CreateVariableHandle("MAIN.Date1")
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub

    Private Sub Form1_Closing(ByVal sender As Object, ByVal e As System.ComponentModel.CancelEventAr
gs) Handles MyBase.Closing
    adsClient.Dispose()
    End Sub


    Private Sub btnRead_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles bt
nRead.Click
    Dim length As Integer
    Dim text As String
    Dim dataStream As AdsStream
    Dim reader As AdsBinaryReader
    Try
        dataStream = New AdsStream(30)
        reader = New AdsBinaryReader(dataStream)
        adsClient.Read(varHandles(0), dataStream)
        textBox1.Text = reader.ReadPlcTIME().ToString()

        dataStream.Position = 0
        adsClient.Read(varHandles(1), dataStream)
        textBox2.Text = reader.ReadPlcDATE().ToString()
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub

    Private Sub btnWrite_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles b
tnWrite.Click
    Dim dataStream As AdsStream
    Dim writer As AdsBinaryWriter
    Dim c As Char
    Try
        dataStream = New AdsStream(4)
        writer = New AdsBinaryWriter(dataStream)
        writer.WritePlcType(TimeSpan.Parse(textBox1.Text))
        adsClient.Write(varHandles(0), dataStream)

        dataStream.Position = 0
        writer.WritePlcType(DateTime.Parse(textBox2.Text))
        adsClient.Write(varHandles(1), dataStream)
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub
End Class
```

PLC program

```
PROGRAM MAIN
VAR
    Time1   :TIME := T#521h33m23s231ms;
    Date1   :DATE := D#1993-06-12;
END_VAR
```

```
;
```

Download

Language / IDE	Unpack the sample program
Visual Basic (for .NET framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12463816331/.exe

6.2.7 Read PLC variable declaration

Download

Language / IDE	Unpack the example program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490142859/.zip
Visual Basic (for .NET Framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490144267/.exe
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490145675/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490147083/.exe

Task

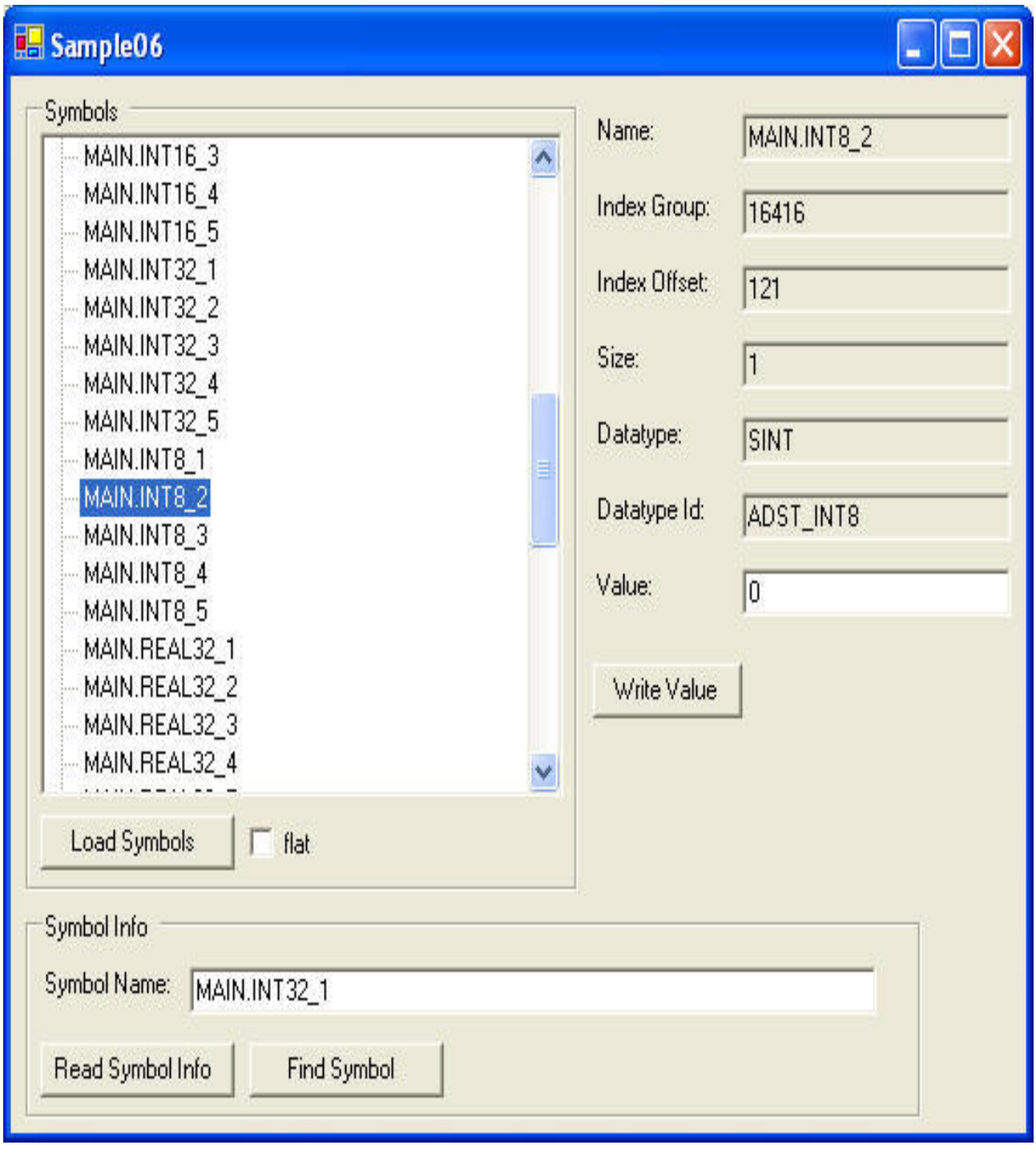
All variables that are declared in the PLC should be displayed in a tree view.

Description

In the Form1_Load method an instance of the TcAdsSymbolInfoLoader [▶ 812] is created with a call to TcAdsClient.CreateSymbolInfoLoader [▶ 693]. This class is responsible for loading the symbol information from the PLC. By clicking the *Load Symbols* button the symbols are loaded by means of the TcAdsSymbolInfoLoader.GetFirstSymbol [▶ 816] method. This method returns the first loaded symbol as a TcAdsSymbolInfo [▶ 780] object. The methods TcAdsSymbolInfo.NextSymbol und TcAdsSymbolInfo.FirstSubSymbol are used to iterate over the symbols and to display the symbols hierarchically in the tree view. If the check box *flat* is checked, the symbols are displayed in a flat list. In this case foreach is used to enumerate over the symbols loaded by the TcAdsSymbolInfoLoader object. This includes the sub symbols.

By selecting a tree view item, the edit boxes are filled with the symbol information. Additionally the value of the variable is read with the help of TcAdClient.ReadSymbol [▶ 722] and is displayed in the value edit box. To write a value to a variable one has to click the *Write Value* button, which leads to a call of TcAdsCient.WriteSymbol [▶ 772].

To read the symbol information for a specific variable, one can either click the *Read Symbol Info* or the *Find Symbol* button. In the first case the method TcAdsClient.ReadSymbolInfo [▶ 724] is called. This leads to an ADS call, to load the information for this symbol from the PLC. In the second case the method TcAdsSymbolInfoLoader.FindSymbol [▶ 814] is called, to search for the symbol in the list of the previous loaded symbols. If no symbols have been loaded before, all symbols are loaded from the PLC.



C# Programm

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using System.IO;
using TwinCAT.Ads;


namespace Sample06
{
    /// <summary>
    /// Summary description for Form1.
    /// </summary>
    public class Form1 : System.Windows.Forms.Form
    {
     private System.Windows.Forms.TreeView treeViewSymbols;
```

```
    private System.Windows.Forms.GroupBox groupBox1;
    private System.Windows.Forms.Button btnLoad;
    private System.Windows.Forms.Label label3;
    private System.Windows.Forms.TextBox tbDatatype;
    private System.Windows.Forms.Label label2;
    private System.Windows.Forms.TextBox tbIndexGroup;
    private System.Windows.Forms.Label label1;
    private System.Windows.Forms.TextBox tbIndexOffset;
    private System.Windows.Forms.Button btnFindSymbol;
    private System.Windows.Forms.Button btnReadSymbolInfo;
    private System.Windows.Forms.GroupBox groupBox2;
    private System.Windows.Forms.Label label4;
    private System.Windows.Forms.Label label5;
    private System.Windows.Forms.TextBox tbDatatypeId;
    private System.Windows.Forms.Label label6;
    private System.Windows.Forms.TextBox tbValue;
    /// <summary>
    /// Required designer variable.
    /// </summary>
    private System.ComponentModel.Container components = null;
    private TcAdsClient adsClient;
    private System.Windows.Forms.TextBox tbSymbolname;
    private System.Windows.Forms.Label label9;
    private System.Windows.Forms.TextBox tbSize;
    private System.Windows.Forms.Button btnWrite;
    private TcAdsSymbolInfoLoader symbolLoader;
    private System.Windows.Forms.TextBox tbName;
    private System.Windows.Forms.Label label7;
    private System.Windows.Forms.CheckBox cbFlat;
    private ITcAdsSymbol currentSymbol = null;

    public Form1()
    {
     InitializeComponent();
    }

    /// <summary>
    /// Clean up any resources being used.
    /// </summary>
    protected override void Dispose( bool disposing )
    {
     if( disposing )
     {
        if (components != null)
        {
       components.Dispose();
        }
     }
     base.Dispose( disposing );
    }

    #region Windows Form Designer generated code
     ...
    #endregion

    /// <summary>
    /// The main entry point for the application.
    /// </summary>
    [STAThread]
    static void Main()
    {
     Application.Run(new Form1());
    }

    private void Form1_Load(object sender, System.EventArgs e)
    {
     try
     {
        adsClient = new TcAdsClient();

       // Connect to local PLC - Runtime 1 - TwinCAT2 Port=801, TwinCAT3 Port=851
       adsClient.Connect(801);

        symbolLoader = adsClient.CreateSymbolInfoLoader();
     }
     catch(Exception err)
     {
        MessageBox.Show(err.Message);
     }
    }
```

```
    private void btnLoad_Click(object sender, System.EventArgs e)
    {
     treeViewSymbols.Nodes.Clear();

     if( !cbFlat.Checked )
     {
        TcAdsSymbolInfo symbol = symbolLoader.GetFirstSymbol(true);
        while( symbol != null )
        {
       treeViewSymbols.Nodes.Add(CreateNewNode(symbol));
       symbol = symbol.NextSymbol;
        }
     }
     else
     {
        foreach( TcAdsSymbolInfo symbol in symbolLoader )
        {
       TreeNode node = new TreeNode(symbol.Name);
       node.Tag = symbol;
       treeViewSymbols.Nodes.Add(node);
        }
     }
    }

    private void btnReadSymbolInfo_Click(object sender, System.EventArgs e)
    {
     try
     {
        ITcAdsSymbol symbol = adsClient.ReadSymbolInfo(tbSymbolname.Text);
        if( symbol == null)
        {
       MessageBox.Show("Symbol " + tbSymbolname.Text + " not found");
       return;
        }
        SetSymbolInfo(symbol);
     }
     catch( Exception err )
     {
        MessageBox.Show(err.Message);
     }
    }

    private void btnFindSymbol_Click(object sender, System.EventArgs e)
    {
     try
     {
        ITcAdsSymbol symbol = symbolLoader.FindSymbol(tbSymbolname.Text);
        if( symbol == null)
        {
       MessageBox.Show("Symbol " + tbSymbolname.Text + " not found");
       return;
        }
        SetSymbolInfo(symbol);
     }
     catch( Exception err )
     {
        MessageBox.Show(err.Message);
     }
    }

    private void btnWrite_Click(object sender, System.EventArgs e)
    {
     try
     {
        if( currentSymbol != null )
       adsClient.WriteSymbol(currentSymbol, tbValue.Text);
     }
     catch(Exception err)
     {
        MessageBox.Show("Unable to write Value. " + err.Message);
     }
    }

    private void treeViewSymbols_AfterSelect(object sender, System.Windows.Forms.TreeViewEventArgs
e)
    {
     if( e.Node.Text.Length > 0 )
     {
        if( e.Node.Tag is TcAdsSymbolInfo )
```

```
            {
           SetSymbolInfo((ITcAdsSymbol)e.Node.Tag);
            }
        }
       }

       private TreeNode CreateNewNode(TcAdsSymbolInfo symbol)
       {
        TreeNode node = new TreeNode(symbol.Name);

        node.Tag = symbol;
        TcAdsSymbolInfo subSymbol = symbol.FirstSubSymbol;
        while( subSymbol != null )
        {
           node.Nodes.Add(CreateNewNode(subSymbol));
           subSymbol = subSymbol.NextSymbol;
        }
        return node;
       }

       private void SetSymbolInfo(ITcAdsSymbol symbol)
       {
        currentSymbol = symbol;
        tbName.Text = symbol.Name.ToString();
        tbIndexGroup.Text = symbol.IndexGroup.ToString();
        tbIndexOffset.Text = symbol.IndexOffset.ToString();
        tbSize.Text = symbol.Size.ToString();
        tbDatatype.Text = symbol.Type;
        tbDatatypeId.Text = symbol.Datatype.ToString();
        try
        {
           tbValue.Text = adsClient.ReadSymbol(symbol).ToString();
        }
        catch( AdsDatatypeNotSupportedException err )
        {
           tbValue.Text = err.Message;
        }
        catch(Exception err)
        {
           MessageBox.Show("Unable to read Symbol Info. " + err.Message);
        }
       }
      }
}
```

PLC program

```
PROGRAM MAIN
VAR
    REAL32_1 AT %MB0 : REAL;   (* 1 *)
    REAL32_2 AT %MB4 : REAL;   (* 2 *)
    REAL32_3 AT %MB8 : REAL;   (* 3 *)
    REAL32_4 AT %MB12: REAL;   (* 4 *)
    REAL32_5 AT %MB16: REAL;   (* 5 *)

    REAL64_1 AT %MB20 : LREAL;   (* 6 *)
    REAL64_2 AT %MB28 : LREAL;   (* 7 *)
    REAL64_3 AT %MB36 : LREAL;   (* 8 *)
    REAL64_4 AT %MB44 : LREAL;   (* 9 *)
    REAL64_5 AT %MB52 : LREAL;   (* 10 *)

    INT32_1 AT %MB60 : DINT;   (* 11 *)
    INT32_2 AT %MB64 : DINT;   (* 12 *)
    INT32_3 AT %MB68 : DINT;   (* 13 *)
    INT32_4 AT %MB72 : DINT;   (* 14 *)
    INT32_5 AT %MB76 : DINT;   (* 15 *)

    UINT32_1 AT %MB80 : UDINT;   (* 16 *)
    UINT32_2 AT %MB84 : UDINT;   (* 17 *)
    UINT32_3 AT %MB88 : UDINT;   (* 18 *)
    UINT32_4 AT %MB92 : UDINT;   (* 19 *)
    UINT32_5 AT %MB96 : UDINT;   (* 20 *)

    INT16_1 AT %MB100 : INT;   (* 21 *)
    INT16_2 AT %MB102 : INT;   (* 22 *)
    INT16_3 AT %MB104 : INT;   (* 23 *)
    INT16_4 AT %MB106 : INT;   (* 24 *)
    INT16_5 AT %MB108 : INT;   (* 25 *)
```

```
    UINT16_1 AT %MB110 : UINT;  (* 26 *)
    UINT16_2 AT %MB112 : UINT;  (* 27 *)
    UINT16_3 AT %MB114 : UINT;  (* 28 *)
    UINT16_4 AT %MB116 : UINT;  (* 29 *)
    UINT16_5 AT %MB118 : UINT;  (* 30 *)

    INT8_1 AT %MB120 : SINT;  (* 31 *)
    INT8_2 AT %MB121 : SINT;  (* 32 *)
    INT8_3 AT %MB122 : SINT;  (* 33 *)
    INT8_4 AT %MB123 : SINT;  (* 34 *)
    INT8_5 AT %MB124 : SINT;  (* 35 *)

    UINT8_1 AT %MB125 : USINT;  (* 36 *)
    UINT8_2 AT %MB126 : USINT;  (* 37 *)
    UINT8_3 AT %MB128 : USINT;  (* 38 *)
    UINT8_4 AT %MB129 : USINT;  (* 39 *)
    UINT8_5 AT %MB130 : USINT;  (* 40 *)

    BOOL_1 AT %MX131.0 : BOOL;  (* 41 *)
    BOOL_2 AT %MX131.1 : BOOL;  (* 42 *)
    BOOL_3 AT %MX131.2 : BOOL;  (* 43 *)
    BOOL_4 AT %MX131.3 : BOOL;  (* 44 *)
    BOOL_5 AT %MX131.4 : BOOL;  (* 45 *)

    ARRAY_1 : ARRAY[1 .. 10] OF SINT; (* 46 *)
    ARRAY_2 : ARRAY[1 .. 10] OF INT;  (* 47 *)
    ARRAY_3 : ARRAY[1 .. 10] OF DINT; (* 48 *)
    ARRAY_4 : ARRAY[1 .. 10] OF LREAL;(* 49 *)
    ARRAY_5 : ARRAY[1 .. 10] OF BOOL; (* 50 *)

    STRING_1 : STRING(20);
END_VAR
```

6.2.7.1 Read PLC variable declaration with Delphi

From TwinCAT.Ads.NET version >= 1.0.0.12

Task

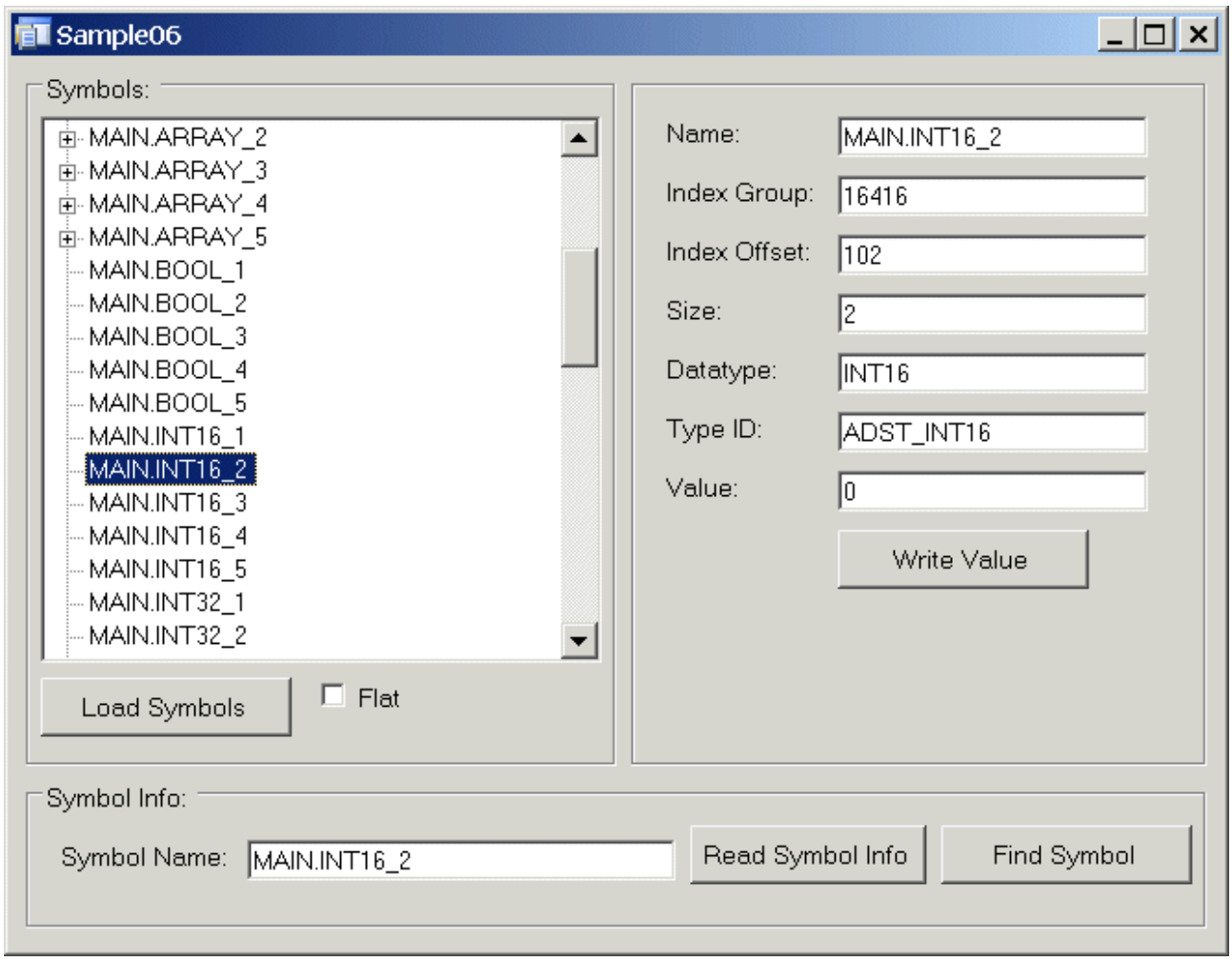
All variables that are declared in the PLC should be displayed in a tree view.

Description

In the*MainForm_Load*method an instance of the TcAdsSymbolInfoLoader [▶ 812] is created with a call to TcAdsClient.CreateSymbolInfoLoader [▶ 693]. This class is responsible for loading the symbol information from the PLC. By clicking the *Load Symbols* button the symbols are loaded by means of the TcAdsSymbolInfoLoader.GetFirstSymbol [▶ 816] method. This method returns the first loaded symbol as a TcAdsSymbolInfo [▶ 780] object. The methods TcAdsSymbolInfo.NextSymbol und TcAdsSymbolInfo.FirstSubSymbol are used to iterate over the symbols and to display the symbols hierarchically in the tree view. If the check box *flat* is checked, the symbols are displayed in a flat list. In this case the application enumerates over the symbols loaded by the TcAdsSymbolInfoLoader object. This includes the sub symbols.

By selecting a tree view item the edit boxes are filled with the symbol information. Additionally the value of the variable is read with the help of TcAdClient.ReadSymbol [▶ 722] and is displayed in the value edit box. To write a value to a variable one has to click the *Write Value* button, which leads to a call of TcAdsCient.WriteSymbol [▶ 772].

To read the symbol information for a specific variable, one can either click the *Read Symbol Info* or the *Find Symbol* button. In the first case the method TcAdsClient.ReadSymbolInfo [▶ 724] is called. This leads to an ADS call, to load the information for this symbol from the PLC. In the second case the method TcAdsSymbolInfoLoader.FindSymbol [▶ 814] is called, to search for the symbol in the list of the previous loaded symbols. If no symbols have been loaded before, all symbols are loaded from the PLC.



Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample06;

interface

uses
  System.Drawing,
  System.Collections,
  System.Collections.Generic,
  System.Windows.Forms,
  System.ComponentModel, System.IO, TwinCAt.Ads;

type
  /// <summary>
  /// Summary description for MainForm.
  /// </summary>
  MainForm = partial class(System.Windows.Forms.Form)
  private
    tcClient      : TcAdsClient;
    symbolLoader  : TcAdsSymbolInfoLoader;
    currentSymbol : ITcAdsSymbol;

    method CreateNewNode(var symbol: TcAdsSymbolInfo) : TreeNode;
    method SetSymbolInfo(symbol: ITcAdsSymbol);
    method MainForm_Load(sender: System.Object; e: System.EventArgs);
    method MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEventArgs)
;
    method btnLoad_Click(sender: System.Object; e: System.EventArgs);
    method btnWrite_Click(sender: System.Object; e: System.EventArgs);
    method btnFindSymbol_Click(sender: System.Object; e: System.EventArgs);
    method btnReadSymbolInfo_Click(sender: System.Object; e: System.EventArgs);
    method treeView1_AfterSelect(sender: System.Object; e: System.Windows.Forms.TreeViewEventArgs);
  protected
    method Dispose(disposing: Boolean); override;
```

```
  public
    constructor;
  end;

implementation

{$REGION Construction and Disposition}
constructor MainForm;
begin
  //
  // Required for Windows Form Designer support
  //
  InitializeComponent();

  //
  // TODO: Add any constructor code after InitializeComponent call
  //
end;

method MainForm.Dispose(disposing: Boolean);
begin
  if disposing then begin
    if assigned(components) then
      components.Dispose();

    //
    // TODO: Add custom disposition code here
    //
  end;
  inherited Dispose(disposing);
end;
{$ENDREGION}

method MainForm.MainForm_Load(sender: System.Object; e: System.EventArgs);
begin
  tcClient := new TcAdsClient();
  tcClient.Connect(801);
  try
    symbolLoader := tcClient.CreateSymbolInfoLoader();
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source);
  end;
end;

method MainForm.MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEvent
Args);
begin
  try
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
  tcClient.Dispose();
end;

method MainForm.btnLoad_Click(sender: System.Object; e: System.EventArgs);
var symbol  : TcAdsSymbolInfo;
    node    : TreeNode;
begin
  treeViewSymbols.Nodes.Clear();

  if not cbFlat.Checked then
  begin
    symbol := symbolLoader.GetFirstSymbol(true);
    while ( symbol <> Nil )do
    begin
      treeViewSymbols.Nodes.Add(CreateNewNode(var symbol));
      symbol := symbol.NextSymbol;
    end;
  end
  else
  begin
    for symbol : TcAdsSymbolInfo in symbolLoader do
    begin
      node := new TreeNode(symbol.Name);
      node.Tag := symbol;
      treeViewSymbols.Nodes.Add(node);
    end;
  end;
```

```
end;

method MainForm.CreateNewNode(var symbol: TcAdsSymbolInfo) : TreeNode;
var node      : TreeNode;
  subSymbol : TcAdsSymbolInfo;
begin
  node := new TreeNode(symbol.Name);
  node.Tag := symbol;
  subSymbol := symbol.FirstSubSymbol;
  while( subSymbol <> Nil ) do
  begin
    node.Nodes.Add(CreateNewNode(var subSymbol));
    subSymbol := subSymbol.NextSymbol;
  end;
  result := node;
end;

method MainForm.SetSymbolInfo(symbol: ITcAdsSymbol);
begin
  currentSymbol := symbol;
  tbSymbolname.Text := symbol.Name;
  tbName.Text := symbol.Name.ToString();
  tbIndexGroup.Text := symbol.IndexGroup.ToString();
  tbIndexOffset.Text := symbol.IndexOffset.ToString();
  tbSize.Text := symbol.Size.ToString();

  tbDatatype.Text := symbol.Type;
  tbDatatypeId.Text := System.Object(symbol.Datatype).ToString();

  try
    tbValue.Text := tcClient.ReadSymbol(symbol).ToString();
  except
    on err: AdsDatatypeNotSupportedException do
      tbValue.Text := err.Message;
    on err: Exception do
      MessageBox.Show(err.Message, err.Source);
  end;
end;

method MainForm.btnWrite_Click(sender: System.Object; e: System.EventArgs);
begin
    try
      if( currentSymbol <> nil )then
    tcClient.WriteSymbol(currentSymbol, tbValue.Text);
    except
      on err: Exception do
    MessageBox.Show('Unable to write Value! ' + err.Message, err.Source);
    end;
end;

method MainForm.btnFindSymbol_Click(sender: System.Object; e: System.EventArgs);
var symbol : ITcAdsSymbol;
begin
  try
    symbol := symbolLoader.FindSymbol(tbSymbolname.Text);
    if( symbol = Nil ) then
      MessageBox.Show('Symbol ' + tbSymbolname.Text + ' not found!', 'TwinCAT.Ads')
    else
      SetSymbolInfo(symbol);
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source);
  end;
end;

method MainForm.btnReadSymbolInfo_Click(sender: System.Object; e: System.EventArgs);
var symbol : ITcAdsSymbol;
begin
  try
    symbol := tcClient.ReadSymbolInfo(tbSymbolname.Text);
    if( symbol = Nil ) then
      MessageBox.Show('Symbol ' + tbSymbolname.Text + ' not found!', 'TwinCAT.Ads')
    else
      SetSymbolInfo(symbol);
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source);
  end;
end;
```

```
method MainForm.treeView1_AfterSelect(sender: System.Object; e: System.Windows.Forms.TreeViewEventAr
gs);
begin
  if( e.Node.Text.Length > 0 ) then
  begin
    if( e.Node.Tag is TcAdsSymbolInfo ) then
    begin
      SetSymbolInfo(ITcAdsSymbol(e.Node.Tag));
    end;
  end;
end;

end.
```

PLC program

```
PROGRAM MAIN
VAR
VAR
    REAL32_1 AT %MB0 : REAL;         (* 1 *)
    REAL32_2 AT %MB4 : REAL;         (* 2 *)
    REAL32_3 AT %MB8 : REAL;         (* 3 *)
    REAL32_4 AT %MB12: REAL;         (* 4 *)
    REAL32_5 AT %MB16: REAL;         (* 5 *)
    REAL64_1 AT %MB20 : LREAL;           (* 6 *)
    REAL64_2 AT %MB28 : LREAL;           (* 7 *)
    REAL64_3 AT %MB36 : LREAL;           (* 8 *)
    REAL64_4 AT %MB44 : LREAL;           (* 9 *)
    REAL64_5 AT %MB52 : LREAL;           (* 10 *)
    INT32_1 AT %MB60 : DINT;         (* 11 *)
    INT32_2 AT %MB64 : DINT;         (* 12 *)
    INT32_3 AT %MB68 : DINT;         (* 13 *)
    INT32_4 AT %MB72 : DINT;         (* 14 *)
    INT32_5 AT %MB76 : DINT;         (* 15 *)
    UINT32_1 AT %MB80 : UDINT;           (* 16 *)
    UINT32_2 AT %MB84 : UDINT;           (* 17 *)
    UINT32_3 AT %MB88 : UDINT;           (* 18 *)
    UINT32_4 AT %MB92 : UDINT;           (* 19 *)
    UINT32_5 AT %MB96 : UDINT;           (* 20 *)
    INT16_1 AT %MB100 : INT;         (* 21 *)
    INT16_2 AT %MB102 : INT;         (* 22 *)
    INT16_3 AT %MB104 : INT;         (* 23 *)
    INT16_4 AT %MB106 : INT;         (* 24 *)
    INT16_5 AT %MB108 : INT;         (* 25 *)
    UINT16_1 AT %MB110 : UINT;           (* 26 *)
    UINT16_2 AT %MB112 : UINT;           (* 27 *)
    UINT16_3 AT %MB114 : UINT;           (* 28 *)
    UINT16_4 AT %MB116 : UINT;           (* 29 *)
    UINT16_5 AT %MB118 : UINT;           (* 30 *)
    INT8_1 AT %MB120 : SINT;         (* 31 *)
    INT8_2 AT %MB121 : SINT;         (* 32 *)
    INT8_3 AT %MB122 : SINT;         (* 33 *)
    INT8_4 AT %MB123 : SINT;         (* 34 *)
    INT8_5 AT %MB124 : SINT;         (* 35 *)
    UINT8_1 AT %MB125 : USINT;           (* 36 *)
    UINT8_2 AT %MB126 : USINT;           (* 37 *)
    UINT8_3 AT %MB128 : USINT;           (* 38 *)
    UINT8_4 AT %MB129 : USINT;           (* 39 *)
    UINT8_5 AT %MB130 : USINT;           (* 40 *)
    BOOL_1 AT %MX131.0 : BOOL;           (* 41 *)
    BOOL_2 AT %MX131.1 : BOOL;           (* 42 *)
    BOOL_3 AT %MX131.2 : BOOL;           (* 43 *)
    BOOL_4 AT %MX131.3 : BOOL;           (* 44 *)
    BOOL_5 AT %MX131.4 : BOOL;           (* 45 *)
    ARRAY_1 : ARRAY[1 .. 10] OF SINT;        (* 46 *)
    ARRAY_2 : ARRAY[1 .. 10] OF INT;     (* 47 *)
    ARRAY_3 : ARRAY[1 .. 10] OF DINT;        (* 48 *)
    ARRAY_4 : ARRAY[1 .. 10] OF LREAL;       (* 49 *)
    ARRAY_5 : ARRAY[1 .. 10] OF BOOL;        (* 50 *)
    STRING_1 : STRING(20);
END_VAR
```

```
;
```

Download

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466768779/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466770187/.exe

6.2.7.2 Read PLC variable declaration with Visual Basic

From TwinCAT.Ads.NET version >= 1.0.0.12

Task

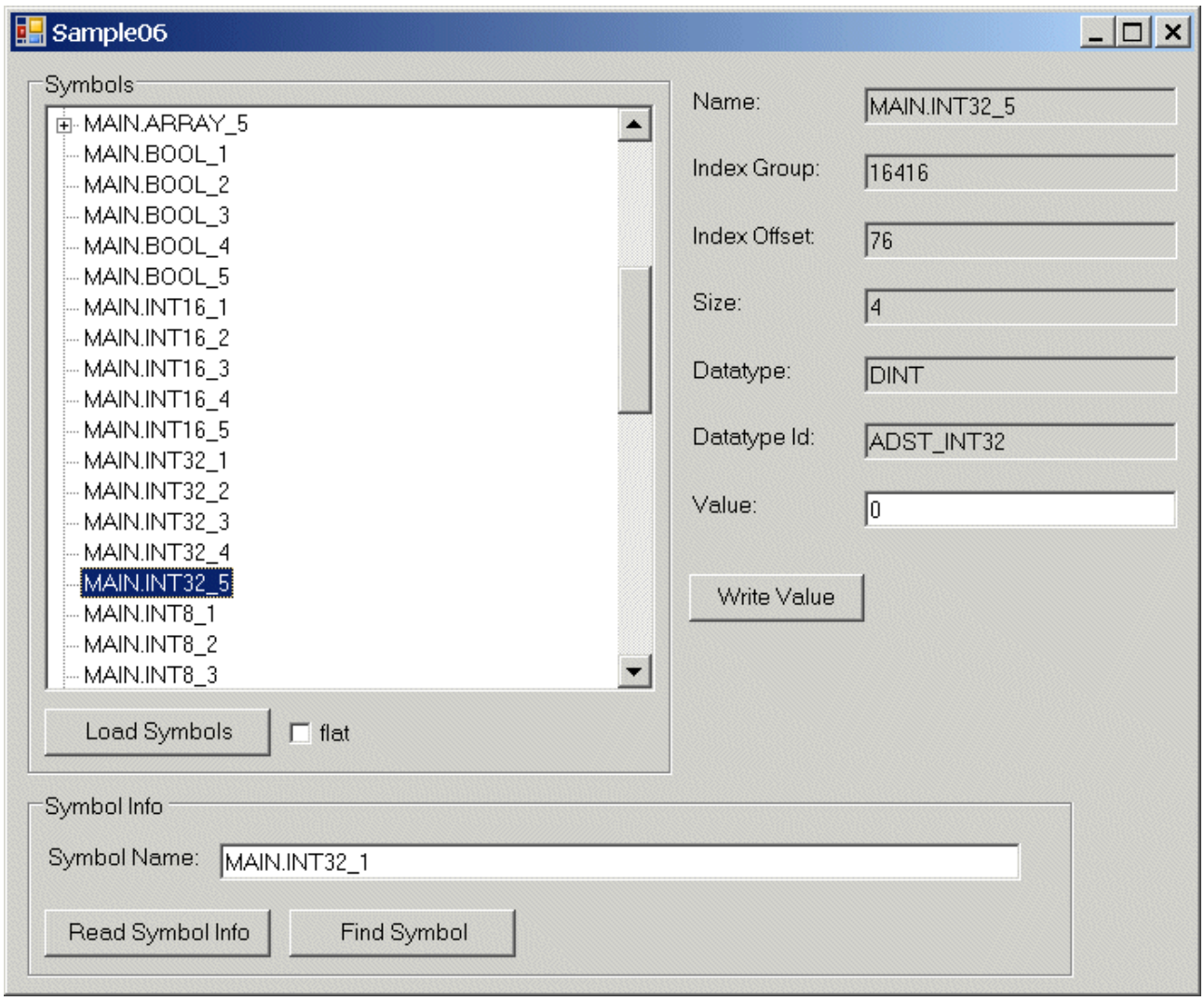
All variables that are declared in the PLC should be displayed in a tree view.

Description

In the*MainForm_Load*method an instance of the TcAdsSymbolInfoLoader [▶ 812] is created with a call to TcAdsClient.CreateSymbolInfoLoader [▶ 693]. This class is responsible for loading the symbol information from the PLC. By clicking the *Load Symbols* button the symbols are loaded by means of the TcAdsSymbolInfoLoader.GetFirstSymbol [▶ 816] method. This method returns the first loaded symbol as a TcAdsSymbolInfo [▶ 780] object. The methods TcAdsSymbolInfo.NextSymbol und TcAdsSymbolInfo.FirstSubSymbol are used to iterate over the symbols and to display the symbols hierarchically in the tree view. If the check box *flat* is checked, the symbols are displayed in a flat list. In this case the application enumerates over the symbols loaded by the TcAdsSymbolInfoLoader object. This includes the sub symbols.

By selecting a tree view item, the edit boxes are filled with the symbol information. Additionally the value of the variable is read with the help of TcAdClient.ReadSymbol [▶ 722] and is displayed in the value edit box. To write a value to a variable one has to click the *Write Value* button, which leads to a call of TcAdsCient.WriteSymbol [▶ 772].

To read the symbol information for a specific variable, one can either click the *Read Symbol Info* or the *Find Symbol* button. In the first case the method TcAdsClient.ReadSymbolInfo [▶ 724] is called. This leads to an ADS call, to load the information for this symbol from the PLC. In the second case the method TcAdsSymbolInfoLoader.FindSymbol [▶ 814] is called, to search for the symbol in the list of the previous loaded symbols. If no symbols have been loaded before, all symbols are loaded from the PLC.



Visual Basic (for .NET framework) program

```
Imports TwinCAT.Ads
Imports System.IO
Public Class Form1
    Inherits System.Windows.Forms.Form

#Region " Windows Form Designer generated code "

...

#End Region

    Private symbolLoader As TcAdsSymbolInfoLoader
    Private adsClient As TcAdsClient
    Private currentSymbol As ITcAdsSymbol = Nothing

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBas
e.Load
    Try
        adsClient = New TcAdsClient
        adsClient.Connect(801)
        symbolLoader = adsClient.CreateSymbolInfoLoader()
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub


    Private Sub btnLoad_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles bt
nLoad.Click
    Dim node As TreeNode
```

```
    treeViewSymbols.Nodes.Clear()
    Try
        If (Not cbFlat.Checked) Then
        Dim symbol As TcAdsSymbolInfo
        symbol = symbolLoader.GetFirstSymbol(True)
        While (Not symbol Is Nothing)
            treeViewSymbols.Nodes.Add(CreateNewNode(symbol))
            symbol = symbol.NextSymbol
        End While
        Else
        For Each symbol As TcAdsSymbolInfo In symbolLoader
            node = New TreeNode(symbol.Name)
            node.Tag = symbol
            treeViewSymbols.Nodes.Add(node)
        Next
        End If
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub

    Private Sub btnReadSymbolInfo_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnReadSymbolInfo.Click
    Dim symbol As ITcAdsSymbol
    Try
        symbol = adsClient.ReadSymbolInfo(tbSymbolname.Text)
        If (symbol Is Nothing) Then
        MessageBox.Show("Symbol " + tbSymbolname.Text + " not found")
        Return
        End If
        Call SetSymbolInfo(symbol)
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub

    Private Sub btnFindSymbol_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Hand
les btnFindSymbol.Click
    Dim symbol As ITcAdsSymbol
    Try
        symbol = symbolLoader.FindSymbol(tbSymbolname.Text)
        If (symbol Is Nothing) Then
        MessageBox.Show("Symbol " + tbSymbolname.Text + " not found")
        Return
        End If
        Call SetSymbolInfo(symbol)
    Catch err As Exception
        MessageBox.Show(err.Message)
    End Try
    End Sub

    Private Sub treeViewSymbols_AfterSelect(ByVal sender As System.Object, ByVal e As System.Windows
.Forms.TreeViewEventArgs) Handles treeViewSymbols.AfterSelect
    If (e.Node.Text.Length > 0) Then
        If (TypeOf e.Node.Tag Is TcAdsSymbolInfo) Then
        SetSymbolInfo(DirectCast(e.Node.Tag, ITcAdsSymbol))
        End If
    End If
    End Sub

    Private Sub btnWrite_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles b
tnWrite.Click
    Try
        If (Not currentSymbol Is Nothing) Then
        adsClient.WriteSymbol(currentSymbol, tbValue.Text)
        End If
    Catch err As Exception
        MessageBox.Show("Unable to write Value. " + err.Message)
    End Try
    End Sub
    Private Function CreateNewNode(ByVal symbol As TcAdsSymbolInfo) As TreeNode
    Dim node As TreeNode
    Dim subsymbol As TcAdsSymbolInfo
    node = New TreeNode(symbol.Name)
    node.Tag = symbol
    subsymbol = symbol.FirstSubSymbol
    While (Not subsymbol Is Nothing)
        node.Nodes.Add(CreateNewNode(subsymbol))
        subsymbol = subsymbol.NextSymbol
    End While
```

```
    Return node
    End Function

    Private Sub SetSymbolInfo(ByVal symbol As ITcAdsSymbol)
    currentSymbol = symbol
    tbName.Text = symbol.Name.ToString()
    tbIndexGroup.Text = symbol.IndexGroup.ToString()
    tbIndexOffset.Text = symbol.IndexOffset.ToString()
    tbSize.Text = symbol.Size.ToString()
    tbDatatype.Text = symbol.Type
    tbDatatypeId.Text = symbol.Datatype.ToString()
    Try
        tbValue.Text = adsClient.ReadSymbol(symbol).ToString()
    Catch err As AdsDatatypeNotSupportedException
        tbValue.Text = err.Message
    Catch err As Exception
        MessageBox.Show("Unable to read Symbol Info. " + err.Message)
    End Try
    End Sub
End Class
```

PLC program

```
PROGRAM MAIN
VAR
VAR
    REAL32_1 AT %MB0 : REAL;         (* 1 *)
    REAL32_2 AT %MB4 : REAL;         (* 2 *)
    REAL32_3 AT %MB8 : REAL;         (* 3 *)
    REAL32_4 AT %MB12: REAL;         (* 4 *)
    REAL32_5 AT %MB16: REAL;         (* 5 *)
    REAL64_1 AT %MB20 : LREAL;           (* 6 *)
    REAL64_2 AT %MB28 : LREAL;           (* 7 *)
    REAL64_3 AT %MB36 : LREAL;           (* 8 *)
    REAL64_4 AT %MB44 : LREAL;           (* 9 *)
    REAL64_5 AT %MB52 : LREAL;           (* 10 *)
    INT32_1 AT %MB60 : DINT;         (* 11 *)
    INT32_2 AT %MB64 : DINT;         (* 12 *)
    INT32_3 AT %MB68 : DINT;         (* 13 *)
    INT32_4 AT %MB72 : DINT;         (* 14 *)
    INT32_5 AT %MB76 : DINT;         (* 15 *)
    UINT32_1 AT %MB80 : UDINT;           (* 16 *)
    UINT32_2 AT %MB84 : UDINT;           (* 17 *)
    UINT32_3 AT %MB88 : UDINT;           (* 18 *)
    UINT32_4 AT %MB92 : UDINT;           (* 19 *)
    UINT32_5 AT %MB96 : UDINT;           (* 20 *)
    INT16_1 AT %MB100 : INT;         (* 21 *)
    INT16_2 AT %MB102 : INT;         (* 22 *)
    INT16_3 AT %MB104 : INT;         (* 23 *)
    INT16_4 AT %MB106 : INT;         (* 24 *)
    INT16_5 AT %MB108 : INT;         (* 25 *)
    UINT16_1 AT %MB110 : UINT;           (* 26 *)
    UINT16_2 AT %MB112 : UINT;           (* 27 *)
    UINT16_3 AT %MB114 : UINT;           (* 28 *)
    UINT16_4 AT %MB116 : UINT;           (* 29 *)
    UINT16_5 AT %MB118 : UINT;           (* 30 *)
    INT8_1 AT %MB120 : SINT;         (* 31 *)
    INT8_2 AT %MB121 : SINT;         (* 32 *)
    INT8_3 AT %MB122 : SINT;         (* 33 *)
    INT8_4 AT %MB123 : SINT;         (* 34 *)
    INT8_5 AT %MB124 : SINT;         (* 35 *)
    UINT8_1 AT %MB125 : USINT;           (* 36 *)
    UINT8_2 AT %MB126 : USINT;           (* 37 *)
    UINT8_3 AT %MB128 : USINT;           (* 38 *)
    UINT8_4 AT %MB129 : USINT;           (* 39 *)
    UINT8_5 AT %MB130 : USINT;           (* 40 *)
    BOOL_1 AT %MX131.0 : BOOL;           (* 41 *)
    BOOL_2 AT %MX131.1 : BOOL;           (* 42 *)
    BOOL_3 AT %MX131.2 : BOOL;           (* 43 *)
    BOOL_4 AT %MX131.3 : BOOL;           (* 44 *)
    BOOL_5 AT %MX131.4 : BOOL;           (* 45 *)
    ARRAY_1 : ARRAY[1 .. 10] OF SINT;        (* 46 *)
    ARRAY_2 : ARRAY[1 .. 10] OF INT;     (* 47 *)
    ARRAY_3 : ARRAY[1 .. 10] OF DINT;        (* 48 *)
    ARRAY_4 : ARRAY[1 .. 10] OF LREAL;       (* 49 *)
```

```
    ARRAY_5 : ARRAY[1 .. 10] OF BOOL;       (* 50 *)
    STRING_1 : STRING(20);
END_VAR
```

```
;
```

Requirements

Language / IDE	Unpack the sample program
Visual Basic (for .NET framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12463817739/.exe

6.2.8 Reading and writing of PLC variables of any type (ReadAny, WriteAny)

Download

Language / IDE	Unpack the example program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490148491/.zip
Visual Basic (for .NET Framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490149899/.exe
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490151307/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490152715/.exe

Task

Read and Write variables of any type with the help of the ReadAny and WriteAny methods.

Description

ReadAny

In the event method btnRead_Click the method TcAdsClient.ReadAny [▶ 714] is used to read a variable by handle:

```
public object ReadAny(int variableHandle, Type type)
public object ReadAny(int variableHandle, Type type, int[] args)
```

The type of the variable is passed to the method in the parameter **type**. In case the method was successfull, the read data will be returned as a object. The type of the object is equal to the type passed in the parameter **type**. Because some data types (arrays and strings) need additional information, an overload of the method ReadAny exists, that takes an additional parameter **args**. E.g., with strings one must pass an integer array of the length 1. Full list of supported types can be found in the documentation of the overloaded method.

Example:

A PLC variable of the type ARRAY[0..3] OF DINT should be read:

```
int hArr;
int[] arr;

hArr = adsClient.CreateVariableHandle(".arr");
arr = (int[]) adsClient.ReadAny(hArr, typeof(int[]), new int[] {4});

...
adsClient.DeleteVariableHandle(hArr);
```

WriteAny

In the event method btnWrite_Click the method TcAdsClient.WriteAny [▶ 763] is used to write to a variable by handle:

```
public void WriteAny(int variableHandle, object value)
public void WriteAny(int variableHandle, object value, int[] args)
```

The parameter **value** is a reference to the object, that should be written to the PLC variable. Full list of supported types of the object **value** can be found in the documentation of the overloaded method.

Example:

A PLC variable of the type ARRAY[0..3] OF DINT should be written:

```
int hArr;
int[] arr = new int[]{1,2,3,4};

hArr = adsClient.CreateVariableHandle(".arr");
adsClient.WriteAny(hArr, arr);

...
adsClient.DeleteVariableHandle(hArr);
```

Reading and writing of structures:

(not possible with the Compact Framework(CE))

To be able to read or write PLC structures the memory layout of the structure or class in .NET must be the same as in the PLC. The layout of a structure or class can be specified with the attribute StructLayoutAttribute. The LayoutKind must be set to LayoutKind.Sequential and the pack must be set to 1. Therefore, the class SimpleStruct is defined as followed:

```
[StructLayout(LayoutKind.Sequential, Pack=1)]
public class SimpleStruct
{
    public double lrealVal;
    public int dintVal1;
}
```

If arrays, strings or boolean values are define the class, one must specify how these fields should be marshalled. This is accomplished with help of the MarshalAs attribute. Because arrays and strings do not have a fixed length in .NET, the property SizeConst is necessary for arrays and strings. It is not possible to marshal multidimensional arrays or arrays of structures with the :NET Framework 1.1. Multidimensional arrays in the PLC must be mapped to one dimensional arrays in .NET.

In the example the MarshalAsAttribute is used in the class ComplexStruct:

```
[StructLayout(LayoutKind.Sequential, Pack=1)]
public class ComplexStruct
{
    public short intVal;
    //specifies how .NET should marshal the array
    //SizeConst specifies the number of elements the array has.
    [MarshalAs(UnmanagedType.ByValArray, SizeConst=4)]
```

```
    public int[] dintArr = new int[4];
    [MarshalAs(UnmanagedType.I1)]
    public bool boolVal;
    public byte byteVal;
    //specifies how .NET should marshal the string
    //SizeConst specifies the number of characters the string has.
    //'(inclusive the terminating null ).
    [MarshalAs(UnmanagedType.ByValTStr, SizeConst=6)]
    public string stringVal = "";
    public SimpleStruct simpleStruct1 =new SimpleStruct();
}
```

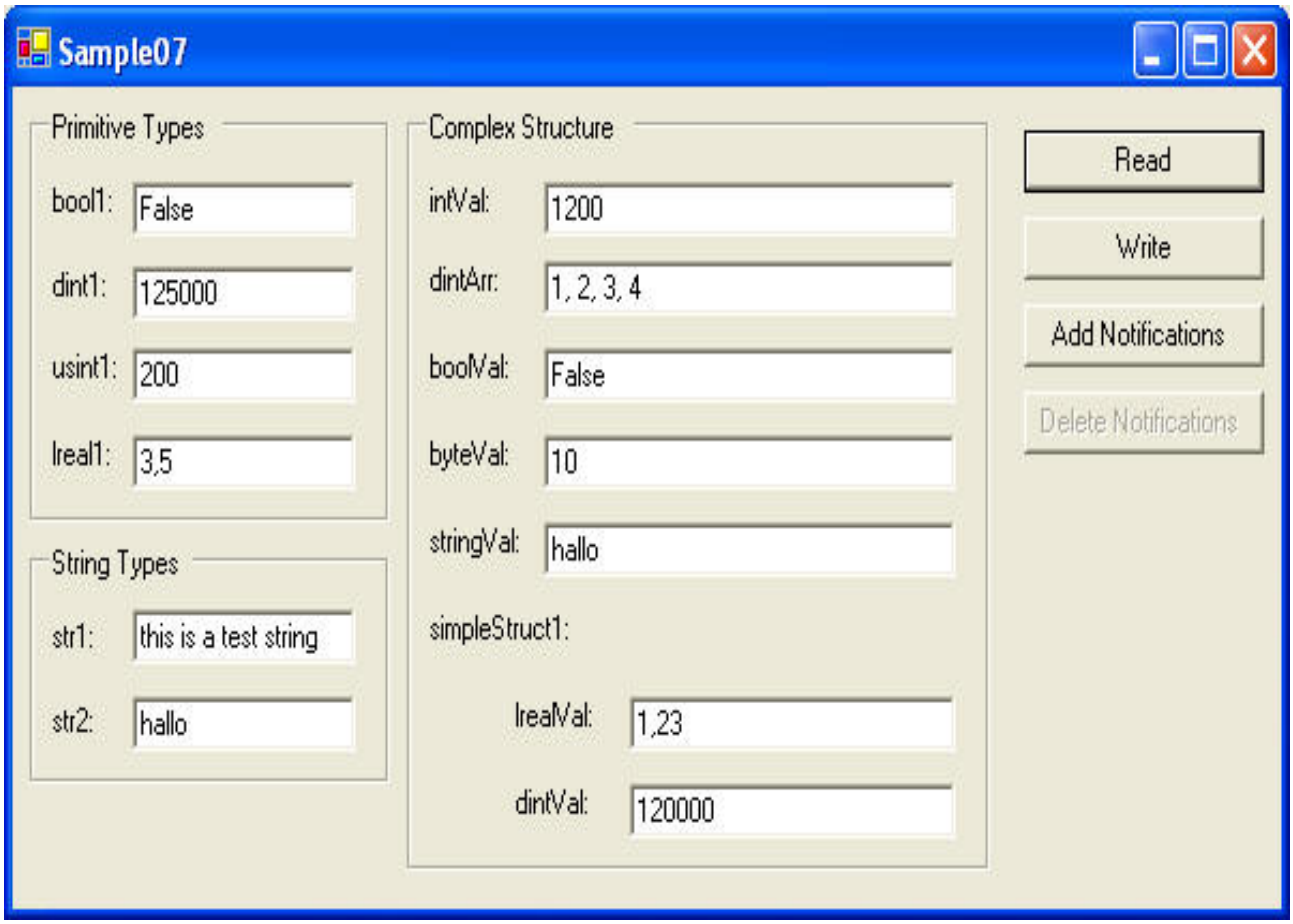
Register ADS notifications

In the event method btnAddNotifications_Click the method AddDeviceNotificationEx [▶ 673] is used to register notifications for a PLC variable. If the value of a variable changes the event AdsNotificationEx [▶ 776] is fired. The difference to the event AdsNotification, is that the value of the variable is stored in an object instead of in an AdsStream. Therefore, one must pass the type of the object to the method AddDeviceNotificationEx:

```
notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.dint1", AdsTransMode.OnChange, 100,
0, tbDint1, typeof(int)));
```

As user object the textbox that should display the value is passed. If the event is fired, the event method **adsClient_AdsNotificationEx** is called. For this the event must be registered in the Form_Load method.

```
adsClient.AdsNotificationEx+=new AdsNotificationExEventHandler(adsClient_AdsNotificationEx);
```



C# program

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using System.Runtime.InteropServices;
using TwinCAT.Ads;


namespace Sample07
{
    /// <summary>
    /// Summary description for Form1.
    /// </summary>
    public class Form1 : System.Windows.Forms.Form
    {
        internal System.Windows.Forms.Button btnDeleteNotifications;
        internal System.Windows.Forms.Button btnAddNotifications;
        internal System.Windows.Forms.Button btnWrite;
        internal System.Windows.Forms.Button btnRead;
        internal System.Windows.Forms.GroupBox GroupBox3;
        internal System.Windows.Forms.TextBox tbComplexStruct_dintArr;
        internal System.Windows.Forms.Label Label14;
        internal System.Windows.Forms.TextBox tbComplexStruct_ByteVal;
        internal System.Windows.Forms.Label Label13;
        internal System.Windows.Forms.TextBox tbComplexStruct_SimpleStruct1_lrealVal;
        internal System.Windows.Forms.Label Label12;
        internal System.Windows.Forms.TextBox tbComplexStruct_SimpleStruct_dintVal;
        internal System.Windows.Forms.Label Label11;
        internal System.Windows.Forms.Label Label5;
        internal System.Windows.Forms.TextBox tbComplexStruct_stringVal;
        internal System.Windows.Forms.Label Label3;
        internal System.Windows.Forms.TextBox tbComplexStruct_boolVal;
        internal System.Windows.Forms.Label Label9;
        internal System.Windows.Forms.TextBox tbComplexStruct_IntVal;
        internal System.Windows.Forms.Label Label10;
        internal System.Windows.Forms.GroupBox GroupBox2;
        internal System.Windows.Forms.TextBox tbStr2;
        internal System.Windows.Forms.Label Label7;
        internal System.Windows.Forms.TextBox tbStr1;
        internal System.Windows.Forms.Label Label8;
        internal System.Windows.Forms.GroupBox GroupBox1;
        internal System.Windows.Forms.TextBox tblreal1;
        internal System.Windows.Forms.Label Label6;
        internal System.Windows.Forms.TextBox tbUsint1;
        internal System.Windows.Forms.Label Label4;
        internal System.Windows.Forms.TextBox tbDint1;
        internal System.Windows.Forms.Label Label2;
        internal System.Windows.Forms.TextBox tbBool1;
        internal System.Windows.Forms.Label Label1;

        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.Container components = null;

        //PLC variable handles
        private int hdint1;
        private int hbool1;
        private int husint1;
        private int hlreal1;
        private int hstr1;
        private int hstr2;
        private int hcomplexStruct;
        private ArrayList notificationHandles;

        private TcAdsClient adsClient;

        public Form1()
        {
            //
            // Required for Windows Form Designer support
            //
            InitializeComponent();
        }

        /// <summary>
```

```
        /// Clean up any resources being used.
        /// </summary>
        protected override void Dispose( bool disposing )
        {
            if( disposing )
            {
                if (components != null)
                {
                    components.Dispose();
                }
            }
            base.Dispose( disposing );
        }

        #region Windows Form Designer generated code
        ..
        #endregion

        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
          Application.Run(new Form1());
        }

        private void Form1_Load(object sender, System.EventArgs e)
        {
            adsClient = new TcAdsClient();
            notificationHandles = new ArrayList();
            try
            {
                // Connect to local PLC - Runtime 1 - TwinCAT2 Port=801, TwinCAT3 Port=851
                adsClient.Connect(801);
                adsClient.AdsNotificationEx+=new AdsNotificationExEventHandler(adsClient_AdsNotifica
tionEx);
                btnDeleteNotifications.Enabled = false;
                //create handles for the PLC variables;
                hbool1 = adsClient.CreateVariableHandle("MAIN.bool1");
                hdint1 = adsClient.CreateVariableHandle("MAIN.dint1");
                husint1 = adsClient.CreateVariableHandle("MAIN.usint1");
                hlreal1 = adsClient.CreateVariableHandle("MAIN.lreal1");
                hstr1 = adsClient.CreateVariableHandle("MAIN.str1");
                hstr2 = adsClient.CreateVariableHandle("MAIN.str2");
                hcomplexStruct = adsClient.CreateVariableHandle("MAIN.ComplexStruct1");
            }
            catch(Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }

        private void Form1_Closing(object sender, System.ComponentModel.CancelEventArgs e)
        {
        adsClient.Dispose();
        }

        private void btnRead_Click(object sender, System.EventArgs e)
        {
            try
            {
                //read by handle
                //the second parameter specifies the type of the variable
                tbDint1.Text = adsClient.ReadAny(hdint1, typeof(int)).ToString();
                tbUsint1.Text = adsClient.ReadAny(husint1, typeof(Byte)).ToString();
                tbBool1.Text = adsClient.ReadAny(hbool1, typeof(Boolean)).ToString();
                tblreal1.Text = adsClient.ReadAny(hlreal1, typeof(Double)).ToString();
                //with strings one has to additionally pass the number of characters
                //specified in the PLC project(default 80).
                //This value is passed is an int array.
                tbStr1.Text = adsClient.ReadAny(hstr1, typeof(String), new int[] {80}).ToString();
                tbStr2.Text = adsClient.ReadAny(hstr2, typeof(String), new int[] {5}).ToString();
                FillStructControls((ComplexStruct)adsClient.ReadAny(hcomplexStruct, typeof(ComplexSt
ruct)));
            }
            catch(Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
```

```
        }

        private void btnWrite_Click(object sender, System.EventArgs e)
        {
            try
            {
                //write by handle
                //the second parameter is the object to be written to the PLC variable
                adsClient.WriteAny(hdint1, int.Parse(tbDint1.Text));
                adsClient.WriteAny(husint1, Byte.Parse(tbUsint1.Text));
                adsClient.WriteAny(hbool1, Boolean.Parse(tbBool1.Text));
                adsClient.WriteAny(hlreal1, Double.Parse(tblreal1.Text));
                //with strings one has to additionally pass the number of characters
                //the variable has in the PLC(default 80).
                adsClient.WriteAny(hstr1, tbStr1.Text, new int[] {80});
                adsClient.WriteAny(hstr2, tbStr2.Text, new int[] {5});
                adsClient.WriteAny(hcomplexStruct, GetStructFromControls()) ;
            }
            catch(Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }

        private void btnAddNotifications_Click(object sender, System.EventArgs e)
        {
            notificationHandles.Clear();
            try
            {
                //register notification
                notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.dint1", AdsTransMode
.OnChange, 100, 0, tbDint1, typeof(int)));
                notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.usint1", AdsTransMod
e.OnChange, 100, 0, tbUsint1, typeof(Byte)));
                notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.bool1", AdsTransMode
.OnChange, 100, 0, tbBool1, typeof(Boolean)));
                notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.lreal1", AdsTransMod
e.OnChange, 100, 0, tblreal1, typeof(Double)));
                notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.str1", AdsTransMode.
OnChange, 100, 0, tbStr1, typeof(String), new int[] {80}));
                notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.str2", AdsTransMode.
OnChange, 100, 0, tbStr2, typeof(String), new int[] {5}));
                notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.complexStruct1", Ads
TransMode.OnChange, 100, 0, tbDint1, typeof(ComplexStruct)));
            }
            catch(Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            btnDeleteNotifications.Enabled = true;
            btnAddNotifications.Enabled = false;
        }

        private void btnDeleteNotifications_Click(object sender, System.EventArgs e)
        {
            //delete registered notifications.
            try
            {
                foreach(int handle in notificationHandles)
                adsClient.DeleteDeviceNotification(handle);
            }
            catch(Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            notificationHandles.Clear();
            btnAddNotifications.Enabled = true;
            btnDeleteNotifications.Enabled = false;
        }

        private void adsClient_AdsNotificationEx(object sender, AdsNotificationExEventArgs e)
        {
            TextBox textBox = (TextBox)e.UserData;
            Type type = e.Value.GetType();
            if(type == typeof(string) || type.IsPrimitive)
                textBox.Text = e.Value.ToString();
            else if(type == typeof(ComplexStruct))
                FillStructControls((ComplexStruct)e.Value);
        }
```

```
        private void FillStructControls(ComplexStruct structure)
        {
            tbComplexStruct_IntVal.Text = structure.intVal.ToString();
            tbComplexStruct_dintArr.Text = String.Format("{0:d}, {1:d}, {2:d}, {3:d}", structure.din
tArr[0],
            structure.dintArr[1], structure.dintArr[2], structure.dintArr[3]);
            tbComplexStruct_boolVal.Text = structure.boolVal.ToString();
            tbComplexStruct_ByteVal.Text = structure.byteVal.ToString();
            tbComplexStruct_stringVal.Text = structure.stringVal;
            tbComplexStruct_SimpleStruct1_lrealVal.Text = structure.simpleStruct1.lrealVal.ToString(
);
            tbComplexStruct_SimpleStruct_dintVal.Text = structure.simpleStruct1.dintVal1.ToString();
        }

        private ComplexStruct GetStructFromControls()
        {
            ComplexStruct structure = new ComplexStruct();
            String[] stringArr = tbComplexStruct_dintArr.Text.Split(new char[] {','});
            structure.intVal = short.Parse(tbComplexStruct_IntVal.Text);
            for(int i=0; i<stringArr.Length; i++)
            structure.dintArr[i] = int.Parse(stringArr[i]);

            structure.boolVal = Boolean.Parse(tbComplexStruct_boolVal.Text);
            structure.byteVal = Byte.Parse(tbComplexStruct_ByteVal.Text);
            structure.stringVal = tbComplexStruct_stringVal.Text;
            structure.simpleStruct1.dintVal1 = int.Parse(tbComplexStruct_SimpleStruct_dintVal.Text);
            structure.simpleStruct1.lrealVal = double.Parse(tbComplexStruct_SimpleStruct1_lrealVal.T
ext);
            return structure;
        }
    }

    // TwinCAT2 Pack = 1, TwinCAT3 Pack = 0
    [StructLayout(LayoutKind.Sequential, Pack=1)]
    public class SimpleStruct
    {
        public double lrealVal;
        public int dintVal1;
    }

    // TwinCAT2 Pack = 1, TwinCAT3 Pack = 0
    [StructLayout(LayoutKind.Sequential, Pack=1)]
    public class ComplexStruct
    {
        public short intVal;
        //specifies how .NET should marshal the array
        //SizeConst specifies the number of elements the array has.
        [MarshalAs(UnmanagedType.ByValArray, SizeConst=4)]
        public int[] dintArr = new int[4];
        [MarshalAs(UnmanagedType.I1)]
        public bool boolVal;
        public byte byteVal;
        //specifies how .NET should marshal the string
        //SizeConst specifies the number of characters the string has.
        //'(inclusive the terminating null ).
        [MarshalAs(UnmanagedType.ByValTStr, SizeConst=6)]
        public string stringVal = "";
        public SimpleStruct simpleStruct1 =new SimpleStruct();
    }
}
```

PLC program

```
TYPE TSimpleStruct :
STRUCT
    lrealVal: LREAL := 1.23;
    dintVal1: DINT := 120000;
END_STRUCT
END_TYPE

TYPE TComplexStruct :
STRUCT
    intVal : INT:=1200;
    dintArr: ARRAY[0..3] OF DINT:= 1,2,3,4;
    boolVal: BOOL := FALSE;
    byteVal: BYTE:=10;
    stringVal : STRING(5) := 'hallo';
    simpleStruct1: TSimpleStruct;
END_STRUCT
```

```
END_TYPE

PROGRAM MAIN
VAR
    (*primitive Types*)
    Bool1:BOOL := FALSE;
    int1:INT := 30000;
    dint1:DINT:=125000;
    usint1:USINT:=200;
    real1:REAL:= 1.2;
    lreal1:LREAL:=3.5;

    (*string Types*)
    str1:STRING := 'this is a test string';
    str2:STRING(5) := 'hallo';

    (*struct Types*)
    complexStruct1 : TComplexStruct;
END_VAR
```

6.2.8.1 Reading and writing of PLC variables of any type with Delphi

From TwinCAT.Ads.NET version >= 1.0.0.15

Task

Read and Write variables of any type with the help of the ReadAny and WriteAny methods.

Description

ReadAny

In the event method btnRead_Click the method TcAdsClient.ReadAny [▶ 714] is used to read a variable by handle:

```
TwinCAT.Ads.TcAdsClient.ReadAny(System.Int32, System.Type)
TwinCAT.Ads.TcAdsClient.ReadAny(System.Int32, System.Type, System.Int32[])
```

The type of the variable is passed to the method in the parameter **type**. In case the method was successfull, the read data will be returned as a object. The type of the object is equal to the type passed in the parameter **type**. Because some data types(arrays and strings) need additional information, an overload of the method ReadAny exists, that takes an additional parameter **args**. Full list of supported types can be found in the documentation of the overloaded method.

Example:

A global PLC variable of the type ARRAY[0..3] OF DINT should be read:

```
var hArr : Integer;
var arr : Arrayof Integer;

hArr := adsClient.CreateVariableHandle(".arr");
arr := Arrayof Integer(adsClient.ReadAny(hArr, TypeOf(Arrayof Integer), [4]));

...
adsClient.DeleteVariableHandle(hArr);
```

WriteAny

In the event method btnWrite_Click the method TcAdsClient.WriteAny [▶ 763] is used to write to a variable by handle:

```
TwinCAT.Ads.TcAdsClient.WriteAny(System.Int32, System.Object)
TwinCAT.Ads.TcAdsClient.WriteAny(System.Int32, System.Object, System.Int32[])
```

The parameter **value** is a reference to the object, that should be written to the PLC variable. Full list of supported types of the object **value** can be found in the documentation of the overloaded method.

Example:

A global PLC variable of the type ARRAY[0..3] OF DINT should be written:

```
var hArr : Integer;
var arr : Arrayof Integer := [1, 2, 3, 4];

hArr := adsClient.CreateVariableHandle(".arr");
adsClient.WriteAny(hArr, arr);

...
adsClient.DeleteVariableHandle(hArr);
```

Reading and writing of structures (not possible with the Compact Framework, CE):

To be able to read or write PLC structures the memory layout of the structure or class in .NET must be the same as in the PLC. The layout of a structure or class can be specified with the attribute StructLayoutAttribute. The LayoutKind must be set to LayoutKind.Sequential and the pack must be set to 1. Therefore the class SimpleStruct is defined as followed:

```
[StructLayout(LayoutKind.Sequential, Pack:=1)]
TSimpleStruct = record
    lrealVal : Double;
    dintVal  : Longint;
end;
```

If arrays, strings or boolean values are define the class, one has to specify how these fields should be marshalled. This is accomplished with help of the MarshalAs attribute. Because arrays and strings do not have a fixed length in .NET, the property SizeConst is necessary for arrays and strings. It is not possible to marshal multidimensional arrays or arrays of structures with the :NET Framework 1.1. Multidimensional arrays in the PLC must be mapped to one dimensional arrays in .NET.

In the example the MarshalAsAttribute is used in the class ComplexStruct:

```
[StructLayout(LayoutKind.Sequential, Pack:=1)]
TComplexStruct = record
  intVal : Smallint;
  // Specifies how .NET should marshal the array
  // SizeConst specifies the number of elements the array has.
  [MarshalAs(UnmanagedType.ByValArray, SizeConst:=4)]
  dintArr : Array[0..3] of Longint;
  [MarshalAs(UnmanagedType.I1)]
  boolVal : Boolean;
  byteVal : Byte;
  // Specifies how .NET should marshal the string
  // SizeConst specifies the number of characters the string has.
  // (inclusive the terminating null ).
[MarshalAs(UnmanagedType.ByValTStr, SizeConst:=6)]
  stringVal : String;
  simpleStruct1 : TSimpleStruct;
end;
```

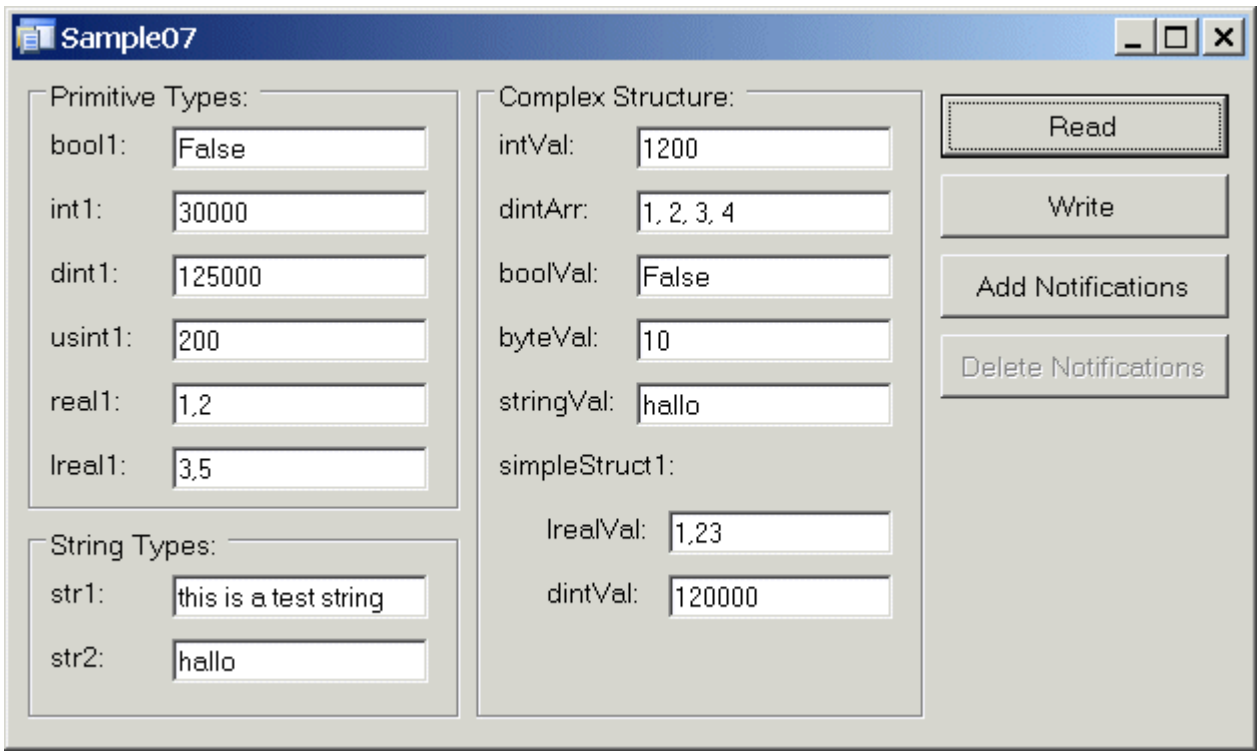
Register ADS notifcations

In the event method btnAddNotifications_Click the method AddDeviceNotificationEx [▶ 673] is used to register notifications for a PLC variable. If the value of a variable changes the event AdsNotificationEx [▶ 776] is fired. The difference to the event AdsNotification, is that the value of the variable is stored in an object instead of in an AdsStream. Therefore one has to pass the type of the object to the method AddDeviceNotificationEx:

```
notificationHandles.Add(adsClient.AddDeviceNotificationEx('MAIN.dint1', AdsTransMode.OnChange, 100,
0, tbDint1, TypeOf(Longint)));
```

As user object the textbox that should display the value is passed. If the event is fired, the event method **adsClient_AdsNotificationEx** is called. For this the event must be registered in the MainForm_Load method.

```
adsClient.AdsNotificationEx += adsClient_AdsNotificationEx;
```



Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample07;

interface

uses
  System.Drawing,
  System.Collections,
  System.Collections.Generic,
  System.Windows.Forms,
  System.ComponentModel, System.IO, TwinCAT.Ads, System.Data, System.Runtime.InteropServices;

type

  ////////////////////////////////////////////////////
  //TYPE TSimpleStruct :
  //STRUCT
  //    lrealVal    : LREAL;
  //    dintVal     : DINT;
  //END_STRUCT
  //END_TYPE
  [StructLayout(LayoutKind.Sequential, Pack:=1)]
  TSimpleStruct = record
    lrealVal : Double;
    dintVal  : Longint;
  end;

  ////////////////////////////////////////////////////
  //TYPE TComplexStruct :
  //STRUCT
  //    intVal      : INT;
  //    dintArr     : ARRAY[0..3] OF DINT;
  //    boolVal     : BOOL;
  //    byteVal     : BYTE;
  //    stringVal       : STRING(5);
  //    simpleStruct1   : TSimpleStruct;
```

```
  //END_STRUCT
  //END_TYPE
  [StructLayout(LayoutKind.Sequential, Pack:=1)]
  TComplexStruct = record
        intVal : Smallint;
      // Specifies how .NET should marshal the array
        // SizeConst specifies the number of elements the array has.
     [MarshalAs(UnmanagedType.ByValArray, SizeConst:=4)]
        dintArr : Array[0..3] of Longint;
        [MarshalAs(UnmanagedType.I1)]
        boolVal : Boolean;
        byteVal : Byte;
      // Specifies how .NET should marshal the string
        // SizeConst specifies the number of characters the string has.
        // (inclusive the terminating null ).
     [MarshalAs(UnmanagedType.ByValTStr, SizeConst:=6)]
        stringVal : String;
    simpleStruct1 : TSimpleStruct;
  end;

  /// <summary>
  /// Summary description for MainForm.
  /// </summary>
  MainForm = partial class(System.Windows.Forms.Form)
  private
    adsClient      : TcAdsClient;
    //PLC variable handles
    hDint1 : Longint;
    hInt1 : Longint;
    hBool1 : Longint;
    hUsint1 : Longint;
    hReal1 : Longint;
    hLreal1 : Longint;
    hStr1 : Longint;
    hStr2 : Longint;
    hComplexStruct : Longint;
    notificationHandles : ArrayList;
    method adsClient_AdsNotificationEx(sender: Object; e: AdsNotificationExEventArgs);
    method FillStructControls(structure : TComplexStruct);
    method GetStructFromControls(var structure : TComplexStruct);
    method MainForm_Load(sender: System.Object; e: System.EventArgs);
    method MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEventArgs)
;
    method btnRead_Click(sender: System.Object; e: System.EventArgs);
    method btnWrite_Click(sender: System.Object; e: System.EventArgs);
    method btnAddNotifications_Click(sender: System.Object; e: System.EventArgs);
    method btnDeleteNotifications_Click(sender: System.Object; e: System.EventArgs);
  protected
    method Dispose(disposing: Boolean); override;
  public
    constructor;
  end;

implementation

{$REGION Construction and Disposition}
constructor MainForm;
begin
  //
  // Required for Windows Form Designer support
  //
  InitializeComponent();

  //
  // TODO: Add any constructor code after InitializeComponent call
  //
end;

method MainForm.Dispose(disposing: Boolean);
begin
  if disposing then begin
    if assigned(components) then
      components.Dispose();

    //
    // TODO: Add custom disposition code here
    //
  end;
  inherited Dispose(disposing);
end;
```

```
{$ENDREGION}

//////////////////////////////////////////////////////////////////////////
//
method MainForm.MainForm_Load(sender: System.Object; e: System.EventArgs);
begin
  try
    // Create instance of TcAdsClient class
    adsClient := new TcAdsClient();
    // Create connection to the local PLC port 801
    adsClient.Connect(801);
    // Create handles for the PLC variables;
    hBool1 := adsClient.CreateVariableHandle('MAIN.bool1');
    hInt1 := adsClient.CreateVariableHandle('MAIN.Int1');
    hDint1 := adsClient.CreateVariableHandle('MAIN.dint1');
    hUsint1 := adsClient.CreateVariableHandle('MAIN.usint1');
    hReal1 := adsClient.CreateVariableHandle('MAIN.real1');
    hLreal1 := adsClient.CreateVariableHandle('MAIN.lreal1');
    hStr1 := adsClient.CreateVariableHandle('MAIN.str1');
    hStr2 := adsClient.CreateVariableHandle('MAIN.str2');
    hComplexStruct := adsClient.CreateVariableHandle('MAIN.ComplexStruct1');
    // Add notification event handler
    adsClient.AdsNotificationEx += adsClient_AdsNotificationEx;

    btnAddNotifications.Enabled := True;
    btnDelNotifications.Enabled := False;
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

//////////////////////////////////////////////////////////////////////////
//
method MainForm.MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEvent
Args);
begin
  try
    // Release handles of PLC variables
    adsClient.DeleteVariableHandle(hBool1);
    adsClient.DeleteVariableHandle(hInt1);
    adsClient.DeleteVariableHandle(hDint1);
    adsClient.DeleteVariableHandle(hUsint1);
    adsClient.DeleteVariableHandle(hReal1);
    adsClient.DeleteVariableHandle(hLreal1);
    adsClient.DeleteVariableHandle(hStr1);
    adsClient.DeleteVariableHandle(hStr2);
    adsClient.DeleteVariableHandle(hComplexStruct);
    // Remove notificaiton event handler
    adsClient.AdsNotificationEx -= adsClient_AdsNotificationEx;
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
  // Close connection
  adsClient.Dispose();
end;

//////////////////////////////////////////////////////////////////////////
//
method MainForm.FillStructControls(structure : TComplexStruct);
begin
  tbIntVal.Text := structure.intVal.ToString();
  tbdintArr.Text := System.String.Format('{0:d}, {1:d}, {2:d}, {3:d}',
  [structure.dintArr[0], structure.dintArr[1], structure.dintArr[2], structure.dintArr[3]]);

  tbBoolVal.Text := structure.boolVal.ToString();
  tbByteVal.Text := structure.byteVal.ToString();
  tbStringVal.Text := structure.stringVal;
  tbLrealVal.Text := structure.simpleStruct1.lrealVal.ToString();
  tbDintVal.Text := structure.simpleStruct1.dintVal.ToString();
end;

//////////////////////////////////////////////////////////////////////////
//
method MainForm.GetStructFromControls(var structure : TComplexStruct);
begin
  structure.intVal := System.Int16.Parse(tbIntVal.Text);
  var sArr := tbDintArr.Text.Split([',']);
  for i : Integer := 0 to 3 do
```

```
    structure.dintArr[i] := System.Int32.Parse(sArr[i]);

  structure.boolVal := System.Boolean.Parse(tbboolVal.Text);
  structure.byteVal := System.Byte.Parse(tbByteVal.Text);
  structure.stringVal := tbStringVal.Text;

  structure.simpleStruct1.dintVal := System.Int32.Parse(tbDintVal.Text);
  structure.simpleStruct1.lrealVal := System.Double.Parse(tbLrealVal.Text);
end;

////////////////////////////////////////////////////////////////////////////
//
method MainForm.btnRead_Click(sender: System.Object; e: System.EventArgs);
begin
  try
    // Read by handle, the second parameter specifies the type of the variable
    tbBool1.Text := adsClient.ReadAny(hBool1, TypeOf(Boolean)).ToString();
    tbInt1.Text := adsClient.ReadAny(hInt1, TypeOf(Smallint)).ToString();
    tbDint1.Text := adsClient.ReadAny(hDint1, TypeOf(Longint)).ToString();
    tbUsint1.Text := adsClient.ReadAny(hUsint1, TypeOf(Byte)).ToString();
    tbReal1.Text := adsClient.ReadAny(hReal1, TypeOf(Single)).ToString();
    tbLreal1.Text := adsClient.ReadAny(hLreal1, TypeOf(Double)).ToString();

    // To read strings we have to pass the number of string characters
    // specified in the PLC project(default 80). This value is passed as an int array.
    tbStr1.Text := adsClient.ReadAny(hstr1, TypeOf(String), [80]).ToString();
    tbStr2.Text := adsClient.ReadAny(hstr2, TypeOf(String), [5]).ToString();

    FillStructControls(TComplexStruct(adsClient.ReadAny(hComplexStruct, TypeOf(TComplexStruct))));
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

////////////////////////////////////////////////////////////////////////////
//
method MainForm.btnWrite_Click(sender: System.Object; e: System.EventArgs);
begin
  try
    var structure := new TComplexStruct();
    // Write by handle, the second parameter is the object to be written to the PLC variable
    adsClient.WriteAny(hBool1, System.Boolean.Parse(tbBool1.Text));
    adsClient.WriteAny(hInt1, System.Int16.Parse(tbInt1.Text));
    adsClient.WriteAny(hDint1, System.Int32.Parse(tbDint1.Text));
    adsClient.WriteAny(hUsint1, System.Byte.Parse(tbUsint1.Text));
    adsClient.WriteAny(hReal1, System.Single.Parse(tbReal1.Text));
    adsClient.WriteAny(hLreal1, System.Double.Parse(tbLreal1.Text));

    // To write strings we have to pass the number of string characters
    // specified in the PLC project(default 80).  This value is passed as an int array.
    adsClient.WriteAny(hStr1, tbStr1.Text, [80]);
    adsClient.WriteAny(hStr2, tbStr2.Text, [5]);

    GetStructFromControls( var structure );
    adsClient.WriteAny(hComplexStruct, structure) ;
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

////////////////////////////////////////////////////////////////////////////
//
method MainForm.btnAddNotifications_Click(sender: System.Object; e: System.EventArgs);
begin
  try
    if (notificationHandles = Nil) then
      notificationhandles := new ArrayList();
    notificationHandles.Clear();

    // Register notification
    notificationHandles.Add(adsClient.AddDeviceNotificationEx('MAIN.bool1', AdsTransMode.OnChange, 1
00, 0, tbBool1, TypeOf(Boolean)));
    notificationHandles.Add(adsClient.AddDeviceNotificationEx('MAIN.int1', AdsTransMode.OnChange, 10
0, 0, tbInt1, TypeOf(Smallint)));
    notificationHandles.Add(adsClient.AddDeviceNotificationEx('MAIN.dint1', AdsTransMode.OnChange, 1
00, 0, tbDint1, TypeOf(Longint)));
    notificationHandles.Add(adsClient.AddDeviceNotificationEx('MAIN.usint1', AdsTransMode.OnChange,
100, 0, tbUsint1, TypeOf(Byte)));
```

```
    notificationHandles.Add(adsClient.AddDeviceNotificationEx('MAIN.real1', AdsTransMode.OnChange, 1
00, 0, tbReal1, TypeOf(Single)));
    notificationHandles.Add(adsClient.AddDeviceNotificationEx('MAIN.lreal1', AdsTransMode.OnChange,
100, 0, tbLreal1, TypeOf(Double)));
    notificationHandles.Add(adsClient.AddDeviceNotificationEx('MAIN.str1', AdsTransMode.OnChange, 10
0, 0, tbStr1, TypeOf(String), [80]));
    notificationHandles.Add(adsClient.AddDeviceNotificationEx('MAIN.str2', AdsTransMode.OnChange, 10
0, 0, tbStr2, TypeOf(String), [5]));
    notificationHandles.Add(adsClient.AddDeviceNotificationEx('MAIN.complexStruct1', AdsTransMode.On
Change, 100, 0, tbDint1, TypeOf(TComplexStruct)));

    btnDelNotifications.Enabled := true;
    btnAddNotifications.Enabled := false;
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

////////////////////////////////////////////////////////////////////////////
//
method MainForm.btnDeleteNotifications_Click(sender: System.Object; e: System.EventArgs);
begin
  try
    btnAddNotifications.Enabled := true;
    btnDelNotifications.Enabled := false;
    // Delete registered notifications.
    for handle : Integer in notificationHandles do
      adsClient.DeleteDeviceNotification(handle);
    notificationHandles.Clear();
  except
    on err: Exception do
      MessageBox.Show(err.Message, err.Source)
  end;
end;

////////////////////////////////////////////////////////////////////////////
//
method MainForm.adsClient_AdsNotificationEx(sender: Object; e: AdsNotificationExEventArgs);
begin
  if( e.Value.GetType() = TypeOf(TComplexStruct))then
    FillStructControls(TComplexStruct(e.Value))
  else
    System.Windows.Forms.TextBox(e.UserData).Text := e.Value.ToString();
end;

end.
```

PLC program

```
TYPE TSimpleStruct :
STRUCT
    lrealVal : LREAL := 1.23;
    dintVal : DINT := 120000;
END_STRUCT
END_TYPE

TYPE TComplexStruct :
STRUCT
    intVal : INT :=1200;
    dintArr : ARRAY[0..3] OF DINT:= 1,2,3,4;
    boolVal : BOOL := FALSE;
    byteVal : BYTE :=10;
    stringVal : STRING(5) := 'hallo';
    simpleStruct1 : TSimpleStruct;
END_STRUCT
END_TYPE

PROGRAM MAIN
VAR
    (*primitive Types*)
    bool1       :BOOL   := FALSE;
    int1        :INT    := 30000;
    dint1       :DINT   :=125000;
    usint1      :USINT  :=200;
    real1       :REAL   := 1.2;
```

```
    lreal1      :LREAL  :=3.5;
    (*string Types*)
    str1        :STRING     := 'this is a test string';
    str2        :STRING(5):= 'hallo';
    (*struct Types*)
    complexStruct1  : TComplexStruct;
END_VAR
;
```

Download

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466771595/.exe
Delphi for .NET (Borland Developer Studio 2006)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466773003/.exe

6.2.8.2 Reading and writing of PLC variables of any type

From TwinCAT.Ads.NET version >= 1.0.0.15

Task

Read and Write variables of any type with the help of the ReadAny and WriteAny methods.

Description

ReadAny

In the event method btnRead_Click the method TcAdsClient.ReadAny [▶ 714] is used to read a variable by handle:

```
TwinCAT.Ads.TcAdsClient.ReadAny(System.Int32, System.Type)
TwinCAT.Ads.TcAdsClient.ReadAny(System.Int32, System.Type, System.Int32[])
```

The type of the variable is passed to the method in the parameter **type**. In case the method was successfully, the read data will be returned as an object. The type of the object is equal to the type passed in the parameter **type**. Because some data types (arrays and strings) need additional information, an overload of the method ReadAny exists, that takes an additional parameter **args**. Full list of supported types can be found in the documentation of the overloaded method.

Example:

A global PLC variable of the type ARRAY[0..3] OF DINT should be read:

```
Dim hArr AsInteger
Dim arr(0 To 3) AsInteger

hArr = adsClient.CreateVariableHandle(".arr")
arr = adsClient.ReadAny(hArr, GetType(Integer()), NewInteger() {4})

...
adsClient.DeleteVariableHandle(hArr)
```

WriteAny

In the event method btnWrite_Click the method TcAdsClient.WriteAny [▶ 763] is used to write to a variable by handle:

```
TwinCAT.Ads.TcAdsClient.WriteAny(System.Int32, System.Object)
TwinCAT.Ads.TcAdsClient.WriteAny(System.Int32, System.Object, System.Int32[])
```

The parameter **value** is a reference to the object, that should be written to the PLC variable. Full list of supported types of the object **value** can be found in the documentation of the overloaded method.

Example:

A global PLC variable of the type ARRAY[0..3] OF DINT should be written:

```
Dim hArr AsInteger
Dim arr AsInteger() = NewInteger() {1, 2, 3, 4}

hArr = adsClient.CreateVariableHandle(".arr")
adsClient.WriteAny(hArr, arr)

...
adsClient.DeleteVariableHandle(hArr)
```

Reading and writing of structures (not possible with the Compact Framework, CE):

To be able to read or write PLC structures the memory layout of the structure or class in .NET must be the same as in the PLC. The layout of a structure or class can be specified with the attribute StructLayoutAttribute. The LayoutKind must be set to LayoutKind.Sequential and the pack must be set to 1. Therefore the class SimpleStruct is defined as followed:

```
<StructLayout(LayoutKind.Sequential, Pack:=1)> _
Public Class SimpleStruct
    Public lrealVal As Double
    Public dintVal1 As Integer
End Class
```

If arrays, strings or boolean values are define the class, one must specify how these fields should be marshalled. This is accomplished with help of the MarshalAs attribute. Because arrays and strings do not have a fixed length in .NET, the property SizeConst is necessary for arrays and strings. It is not possible to marshal multidimensional arrays or arrays of structures with the :NET Framework 1.1. Multidimensional arrays in the PLC must be mapped to one dimensional arrays in .NET.

In the example the MarshalAsAttribute is used in the class ComplexStruct:

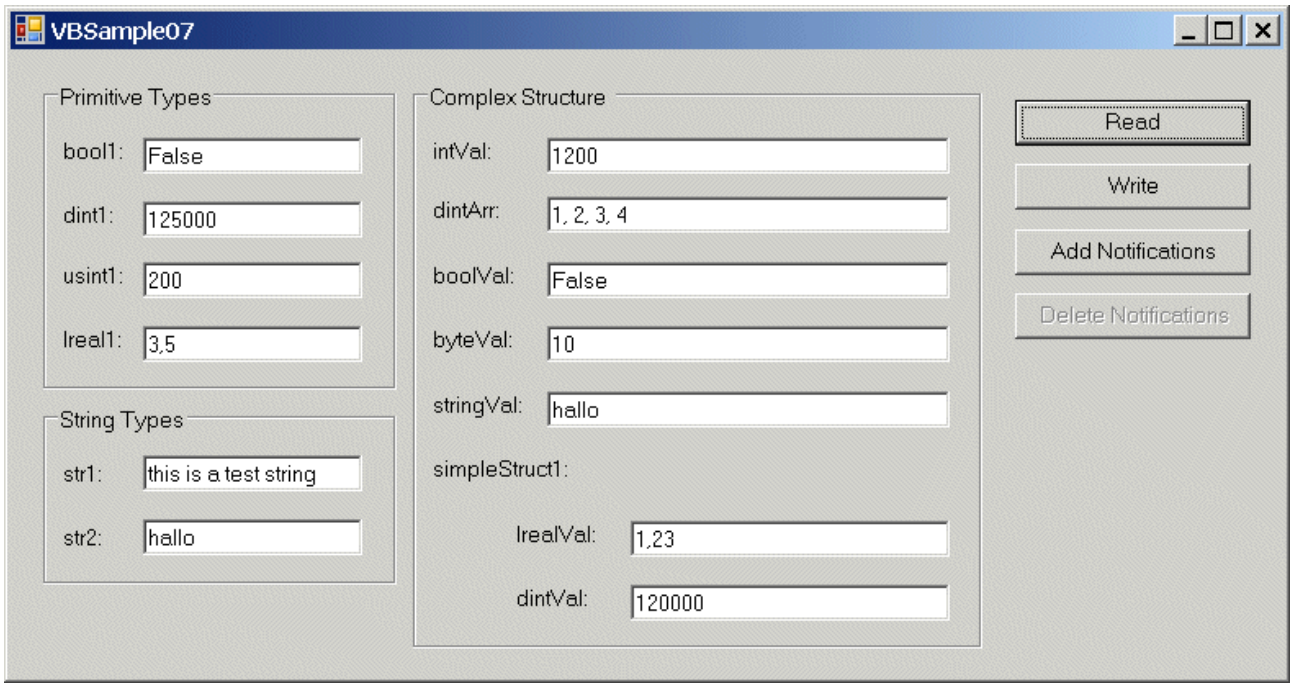
```
<StructLayout(LayoutKind.Sequential, Pack:=1)> _
Public Class ComplexStruct
    Public intVal As Short
    'specifies how .NET should marshal the array
    'SizeConst specifies the number of elements the array has.
    <MarshalAs(UnmanagedType.ByValArray, SizeConst:=4)> _
    Public dintArr(4) As Integer
    <MarshalAs(UnmanagedType.I1)> _
    Public boolVal As Boolean
    Public byteVal As Byte
    'specifies how .NET should marshal the string
    'SizeConst specifies the number of characters the string has.
    '(inclusive the terminating null ).
    <MarshalAs(UnmanagedType.ByValTStr, SizeConst:=6)> _
    Public stringVal As String = ""
    Public simpleStruct1 As SimpleStruct = New SimpleStruct
End Class
```

Register ADS notifcations

In the event method btnAddNotifications_Click the method AddDeviceNotificationEx [▶ 673] is used to register notifications for a PLC variable. If the value of a variable changes the event AdsNotificationEx [▶ 776] is fired. The difference to the event AdsNotification, is that the value of the variable is stored in an object instead of in an AdsStream. Therefore one has to pass the type of the object to the method AddDeviceNotificationEx:

```
notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.dint1", AdsTransMode.OnChange, 100,
0, tbDint1, GetType(Integer)))
```

As user object the textbox that should display the value is passed. If the event is fired, the event method **adsClient_AdsNotificationEx** is called.



Visual Basic (for .NET framework) program

```
Imports TwinCAT.Ads
Imports System.Runtime.InteropServices

Public Class Form1
    Inherits System.Windows.Forms.Form

    'PLC variable handles
    Private hdint1 As Integer
    Private hbool1 As Integer
    Private husint1 As Integer
    Private hlreal1 As Integer
    Private hstr1 As Integer
    Private hstr2 As Integer
    Private hcomplexStruct As Integer
    Private notificationHandles As ArrayList

#Region " Windows Form Designer generated code "

...

#End Region
    Private WithEvents adsClient As TcAdsClient

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBas
e.Load
    adsClient = New TcAdsClient
    notificationHandles = New ArrayList
    Try
        adsClient.Connect(801)
        btnDeleteNotifications.Enabled = False
        'create handles for the PLC variables
        hbool1 = adsClient.CreateVariableHandle("MAIN.bool1")
        hdint1 = adsClient.CreateVariableHandle("MAIN.dint1")
        husint1 = adsClient.CreateVariableHandle("MAIN.usint1")
        hlreal1 = adsClient.CreateVariableHandle("MAIN.lreal1")
        hstr1 = adsClient.CreateVariableHandle("MAIN.str1")
        hstr2 = adsClient.CreateVariableHandle("MAIN.str2")
        hcomplexStruct = adsClient.CreateVariableHandle("MAIN.ComplexStruct1")
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
```

```
    End Sub

    Private Sub Form1_Closing(ByVal sender As Object, ByVal e As System.ComponentModel.CancelEventAr
gs) Handles MyBase.Closing
    adsClient.Dispose()
    End Sub

    Private Sub btnRead_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles bt
nRead.Click
    Try
        'read by handle
        'the second parameter specifies the type of the variable
        tbDint1.Text = adsClient.ReadAny(hdint1, GetType(Integer)).ToString()
        tbUsint1.Text = adsClient.ReadAny(husint1, GetType(Byte)).ToString()
        tbBool1.Text = adsClient.ReadAny(hbool1, GetType(Boolean)).ToString()
        tblreal1.Text = adsClient.ReadAny(hlreal1, GetType(Double)).ToString()
        'with strings one has to additionally pass the number of characters
        'specified in the PLC project(default 80).
        'This value is passed is an Integer array.
        tbStr1.Text = adsClient.ReadAny(hstr1, GetType(String), New Integer() {80}).ToString()
        tbStr2.Text = adsClient.ReadAny(hstr2, GetType(String), New Integer() {5}).ToString()
        FillStructControls(adsClient.ReadAny(hcomplexStruct, GetType(ComplexStruct)))
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
    End Sub

    Private Sub btnWrite_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles b
tnWrite.Click
    Try
        'write by handle
        'the second parameter is the object to be written to the PLC variable
        adsClient.WriteAny(hdint1, Integer.Parse(tbDint1.Text))
        adsClient.WriteAny(husint1, Byte.Parse(tbUsint1.Text))
        adsClient.WriteAny(hbool1, Boolean.Parse(tbBool1.Text))
        adsClient.WriteAny(hlreal1, Double.Parse(tblreal1.Text))
        'with strings one has to additionally pass the number of characters
        'the variable has in the PLC(default 80).
        adsClient.WriteAny(hstr1, tbStr1.Text, New Integer() {80})
        adsClient.WriteAny(hstr2, tbStr2.Text, New Integer() {5})
        adsClient.WriteAny(hcomplexStruct, GetStructFromControls())
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
    End Sub

    Private Sub btnAddNotifications_Click(ByVal sender As System.Object, ByVal e As System.EventArgs
) Handles btnAddNotifications.Click
    notificationHandles.Clear()
    Try
        'register notification
        notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.dint1", AdsTransMode.OnChang
e, 100, 0, tbDint1, GetType(Integer)))
        notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.usint1", AdsTransMode.OnChan
ge, 100, 0, tbUsint1, GetType(Byte)))
        notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.bool1", AdsTransMode.OnChang
e, 100, 0, tbBool1, GetType(Boolean)))
        notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.lreal1", AdsTransMode.OnChan
ge, 100, 0, tblreal1, GetType(Double)))
        notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.str1", AdsTransMode.OnChange
, 100, 0, tbStr1, GetType(String), New Integer() {80}))
        notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.str2", AdsTransMode.OnChange
, 100, 0, tbStr2, GetType(String), New Integer() {5}))
        notificationHandles.Add(adsClient.AddDeviceNotificationEx("MAIN.complexStruct1", AdsTransMod
e.OnChange, 100, 0, tbDint1, GetType(ComplexStruct)))
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
    btnDeleteNotifications.Enabled = True
    btnAddNotifications.Enabled = False
    End Sub

    Private Sub btnDeleteNotifications_Click(ByVal sender As System.Object, ByVal e As System.EventA
rgs) Handles btnDeleteNotifications.Click
    'delete registered notifications.
    Try
        For Each handle As Integer In notificationHandles
        adsClient.DeleteDeviceNotification(handle)
        Next
    Catch ex As Exception
```

```
        MessageBox.Show(ex.Message)
    End Try
    notificationHandles.Clear()
    btnAddNotifications.Enabled = True
    btnDeleteNotifications.Enabled = False
    End Sub

    Private Sub adsClient_AdsNotificationEx(ByVal sender As Object, ByVal e As TwinCAT.Ads.AdsNotifi
cationExEventArgs) Handles adsClient.AdsNotificationEx
    Dim txtBox As TextBox = e.UserData
    Dim type As Type = e.Value.GetType()
    If (type Is GetType(String) Or type.IsPrimitive()) Then
        txtBox.Text = e.Value
    ElseIf (type Is GetType(ComplexStruct)) Then
        FillStructControls(DirectCast(e.Value, ComplexStruct))
    End If
    End Sub

    Public Sub FillStructControls(ByVal struct As ComplexStruct)
    tbComplexStruct_IntVal.Text = struct.intVal
    tbComplexStruct_dintArr.Text = String.Format("{0}, {1}, {2}, {3}", struct.dintArr(0), struct.din
tArr(1), struct.dintArr(2), struct.dintArr(3))
    tbComplexStruct_boolVal.Text = struct.boolVal
    tbComplexStruct_ByteVal.Text = struct.byteVal
    tbComplexStruct_stringVal.Text = struct.stringVal
    tbComplexStruct_SimpleStruct1_lrealVal.Text = struct.simpleStruct1.lrealVal
    tbComplexStruct_SimpleStruct_dintVal.Text = struct.simpleStruct1.dintVal1
    End Sub

    Public Function GetStructFromControls() As ComplexStruct
    Dim struct As ComplexStruct = New ComplexStruct
    Dim stringArr() As String = tbComplexStruct_dintArr.Text.Split(New Char() {","})

    struct.intVal = Short.Parse(tbComplexStruct_IntVal.Text)
    For i As Integer = 0 To stringArr.Length - 1
        struct.dintArr(i) = Integer.Parse(stringArr(i))
    Next
    struct.boolVal = Boolean.Parse(tbComplexStruct_boolVal.Text)
    struct.byteVal = Byte.Parse(tbComplexStruct_ByteVal.Text)
    struct.stringVal = tbComplexStruct_stringVal.Text
    struct.simpleStruct1.dintVal1 = Integer.Parse(tbComplexStruct_SimpleStruct_dintVal.Text)
    struct.simpleStruct1.lrealVal = Double.Parse(tbComplexStruct_SimpleStruct1_lrealVal.Text)
    Return struct
    End Function
End Class

<StructLayout(LayoutKind.Sequential, Pack:=1)> _
Public Class SimpleStruct
    Public lrealVal As Double
    Public dintVal1 As Integer
End Class

<StructLayout(LayoutKind.Sequential, Pack:=1)> _
Public Class ComplexStruct
    Public intVal As Short
    'specifies how .NET should marshal the array
    'SizeConst specifies the number of elements the array has.
    <MarshalAs(UnmanagedType.ByValArray, SizeConst:=4)> _
    Public dintArr(4) As Integer
    <MarshalAs(UnmanagedType.I1)> _
    Public boolVal As Boolean
    Public byteVal As Byte
    'specifies how .NET should marshal the string
    'SizeConst specifies the number of characters the string has.
    '(inclusive the terminating null ).
    <MarshalAs(UnmanagedType.ByValTStr, SizeConst:=6)> _
    Public stringVal As String = ""
    Public simpleStruct1 As SimpleStruct = New SimpleStruct
End Class
```

PLC program

```
TYPE TSimpleStruct :
STRUCT
    lrealVal : LREAL := 1.23;
    dintVal : DINT := 120000;
END_STRUCT
END_TYPE
```

```
TYPE TComplexStruct :
STRUCT
    intVal : INT :=1200;
    dintArr : ARRAY[0..3] OF DINT:= 1,2,3,4;
    boolVal : BOOL := FALSE;
    byteVal : BYTE :=10;
    stringVal : STRING(5) := 'hallo';
    simpleStruct1 : TSimpleStruct;
END_STRUCT
END_TYPE

PROGRAM MAIN
VAR
    (*primitive Types*)
    bool1       :BOOL   := FALSE;
    int1        :INT    := 30000;
    dint1       :DINT   :=125000;
    usint1      :USINT  :=200;
    real1       :REAL   := 1.2;
    lreal1      :LREAL  :=3.5;
    (*string Types*)
    str1        :STRING     := 'this is a test string';
    str2        :STRING(5):= 'hallo';
    (*struct Types*)
    complexStruct1  : TComplexStruct;
END_VAR
```

```
;
```

Download

Language / IDE	Unpack the sample program
Visual Basic (for .NET framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12463819147/.exe

6.2.9 Detect state changes in TwinCAT router and PLC

Download

Language / IDE	Extract the sample program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490154123/.zip
Visual Basic (for .NET Framework)	-
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490155531/.exe
Delphi for .NET (Borland Developer Studio 2006)	-

Task

Detect state changes in TwinCAT router and PLC.

Description

State changes in ADS devices can be detected effectively by registering callback functions to the devices. These functions are called on state changes of the ADS devices.

State changes in the TwinCAT router can be detected via the TcAdsClient.AmsRouterNotification [▶ 779] in the TcAdsClient [▶ 625] class . To detect state changes in the PLC an ADS Notification to the ADS status word must be registered. Callback functions can be registered for both. These are called on state changes of the devices.

The following program monitors the state of the TwinCAT router and the PLC using the above techniques.

C# Program

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.IO;
using TwinCAT.Ads;

namespace TwinCATAds_Sample08
{
  public partial class Form1 : Form
  {
    private TcAdsClient _tcClient = null;
    private AdsStream _adsStream = null;
    private BinaryReader _binRead = null;
    private int _notificationHandle = 0;

    public Form1()
    {
      InitializeComponent();
    }

    private void Form1_Load(object sender, EventArgs e)
    {
      try
      {
    _tcClient = new TcAdsClient();


    // Connect to local PLC - Runtime 1 - TwinCAT2 Port=801, TwinCAT3 Port=851
    _tcClient.Connect(801);

    _adsStream = new AdsStream(2); /* stream for storing the ADS state of the PLC */
    _binRead = new BinaryReader(_adsStream); /* reader for reading the state */

    /* register callback function to detect state changes in the router  */
    _tcClient.AmsRouterNotification += new AmsRouterNotificationEventHandler(AmsRouterNotificationCa
llback);

    /* register an ADS notification an the ADS status word of the PLC  */
    _notificationHandle = _tcClient.AddDeviceNotification(
                (int)AdsReservedIndexGroups.DeviceData, /* index group of the device state*/
                (int)AdsReservedIndexOffsets.DeviceDataAdsState, /
*index offset of the device state */
                _adsStream, /* stream to store the state */
                AdsTransMode.OnChange, /* transfer mode: transmit ste on change */
                0, /* transmit changes immediately */
                0,
                null);

    /* register callback function to react on notifications */
    _tcClient.AdsNotification += new AdsNotificationEventHandler(OnAdsNotification);
      }
    catch (AdsErrorException ex)
      {
    MessageBox.Show(ex.Message);
      }
    }

    /* callback function called on state changes of the router */
    void OnAdsNotification(object sender, AdsNotificationEventArgs e)
    {
      if (e.NotificationHandle == _notificationHandle)
      {
    AdsState plcState = (AdsState)_binRead.ReadInt16(); /* state was written to the stream */
       _plcLabelValue.Text = plcState.ToString();
      }
    }

    /* Ccallback function called on state changes of the PLC */
    void AmsRouterNotificationCallback(object sender, AmsRouterNotificationEventArgs e)
    {
      _routerLabelValue.Text = e.State.ToString();
    }

    private void _exitButton_Click(object sender, EventArgs e)
```

```
    {
      this.Close();
    }

    private void Form1_FormClosing(object sender, FormClosingEventArgs e)
    {
      try
      {
    _tcClient.DeleteDeviceNotification(_notificationHandle);
    _tcClient.Dispose();
      }
      catch(AdsErrorException ex)
      {
    MessageBox.Show(ex.Message);
      }
    }
  }
}
```

6.2.9.1 Detect state changes in TwinCAT router and PLC with Delphi

From TwinCAT.Ads.NET version >= 1.0.0.15

Task

Detect state changes in TwinCAT router and TwinCAT PLC.

Description

State changes in ADS devices can be detected effectively by registering callback functions to the devices. These functions are called on state changes of the ADS devices.

State changes in the TwinCAT router can be detected via the TcAdsClient.AmsRouterNotification [▶ 779] in the TcAdsClient [▶ 625] class . To detect state changes in the TwinCAT-PLC an ADS Notification to the ADS status word must be registered. Callback functions can be registered for both. These are called on state changes of the devices.

Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample08;

interface

uses
  System.Drawing,
  System.Collections,
  System.Collections.Generic,
  System.Windows.Forms,
  System.ComponentModel,
  System.IO,
  TwinCAT.Ads;

type
  /// <summary>
  /// Summary description for MainForm.
  /// </summary>
  MainForm = partial class(System.Windows.Forms.Form)
  private
    tcClient:TcAdsClient;
```

```
    adsStream : AdsStream;
    binRead : BinaryReader;
    hNotification : Int32;

    method tcClient_AdsNotification(sender: Object; e: AdsNotificationEventArgs);
    method tcClient_AmsRouterNotification(sender: Object; e: AmsRouterNotificationEventArgs);
    method MainForm_Load(sender: System.Object; e: System.EventArgs);
    method MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEventArgs)
;
  protected
    method Dispose(disposing: Boolean); override;
  public
    constructor;
  end;

implementation

{$REGION Construction and Disposition}
constructor MainForm;
begin
  //
  // Required for Windows Form Designer support
  //
  InitializeComponent();

  //
  // TODO: Add any constructor code after InitializeComponent call
  //
end;

method MainForm.Dispose(disposing: Boolean);
begin
  if disposing then begin
    if assigned(components) then
      components.Dispose();

    //
    // TODO: Add custom disposition code here
    //
  end;
  inherited Dispose(disposing);
end;
{$ENDREGION}

////////////////////////////////////////////////////////////////////////////////////////////////////
/////////
method MainForm.MainForm_Load(sender: System.Object; e: System.EventArgs);
begin
  try
    // Create instance
    tcClient := new TcAdsClient();
    // Connect the client to the local PLC
    tcClient.Connect(801);

    adsStream := new AdsStream(2);          // Stream storing the ADS state of the PLC
    binRead := new BinaryReader(adsStream);      // Reader to read the state data
    // Add TwinCAT Router notification event handler
    tcClient.AmsRouterNotification += tcClient_AmsRouterNotification;

    // Add TwinCAT PLC device notification
    hNotification := tcClient.AddDeviceNotification(
          Int32( AdsReservedIndexGroups.DeviceData ), // Index group of the device state
        Int32( AdsReservedIndexOffsets.DeviceDataAdsState ), // Index offsset of the device state
        adsStream,// Stream to store the state
        AdsTransMode.OnChange,// Transfer mode: transmit state on change
        0,// Transmit changes immediately
        0, Nil);

    // Register callback to react on state changes of the local PLC
    tcClient.AdsNotification += tcClient_AdsNotification

  except
    On err : AdsErrorException do
       MessageBox.Show(err.Message, err.Source)
  end;
end;

////////////////////////////////////////////////////////////////////////////////////////////////////
/////////
method MainForm.MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEvent
```

```
Args);
begin
  try
    // Remove TwinCAT PLC notification event handler
    tcClient.DeleteDeviceNotification(hNotification);
    tcClient.AdsNotification -= tcClient_AdsNotification;
    // Remove TwinCAT Router notification event handler
    tcClient.AmsRouterNotification -= tcClient_AmsRouterNotification;
    // Close connection
    tcClient.Dispose()
  except
    On err : AdsErrorException do
       MessageBox.Show(err.Message, err.Source)
  end;
end;

////////////////////////////////////////////////////////////////////////////////////////////////////
/////////
method MainForm.tcClient_AmsRouterNotification(sender: Object; e: AmsRouterNotificationEventArgs);
begin
   txtRouterState.Text := e.State.ToString();
end;

////////////////////////////////////////////////////////////////////////////////////////////////////
/////////
method MainForm.tcClient_AdsNotification(sender: Object; e: AdsNotificationEventArgs);
begin
  if (e.NotificationHandle = hNotification) then
    txtPLCState.Text := AdsState(binRead.ReadInt16()).ToString();
end;

end.
```

Download

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466774411/.exe

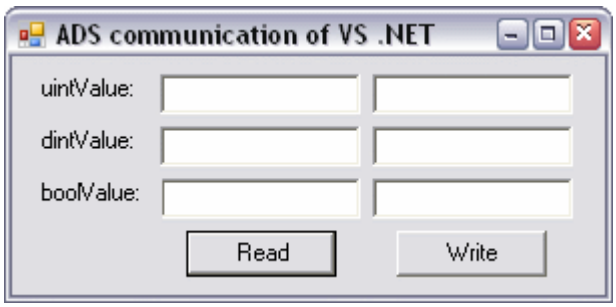
6.2.10 ADS-Sum Command: Reading or writing several variables

Requirement: TwinCAT 2.11 >= Build 1550

Download

Language / IDE	Extract the sample program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490156939/.zip
Visual Basic (for .NET Framework)	-
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490158347/.exe
Delphi for .NET (Borland Developer Studio 2006)	-

Using the ADS Sum Command it is possible to read or write several variables in one command. Designed as TcAdsClient.ReadWrite it is used as a container, which transports all sub-commands in one ADS stream.



In the beginning it's important, binding *'TwinCAT.Ads.dll'* to your project! To do so, open *'Solution Explorer'* and choose *'Add References...'* via*'References'*. *'Browse'* for the DLL in the folder *'TwinCAT -> ADS API -> .NET'* .

C# program

1. Read variables

First, define two structures:

```
namespace AdsBlockRead
{
// Structure declaration for values
  internal struct MyStruct
  {
    public ushort uintValue;
    public int dintValue;
    public bool boolValue;
  }

// Structure declaration for handles
  internal struct VariableInfo
  {
    public int indexGroup;
    public int indexOffset;
    public int length;
  }
```

and declare some global variables.

```
public classForm1 : System.Windows.Forms.Form
    {
    [...]

    privateTcAdsClient adsClient;
    private string[] variableNames;
    private int[] variableLengths;
    VariableInfo[] variables;
```

On applications start an ADS connection to PLC is established and Handle parameters are written into the structure.

```
private void Form1_Load(object sender, System.EventArgs e)
    {
        try
        {
        // Connect to PLC
        adsClient = newTcAdsClient();
        adsClient.Connect(801);

        // Fill structures with name and size of PLC variables
        variableNames = new string[] { "MAIN.uintValue", "MAIN.dintValue", "MAIN.boolValue" };
        variableLengths = new int[] { 2, 4, 1 };

        // Write handle parameter into structure
        variables = newVariableInfo[variableNames.Length];
```

```
        for (int i = 0; i < variables.Length; i++)
        {
            variables[i].indexGroup = (int)AdsReservedIndexGroups.SymbolValueByHandle;
            variables[i].indexOffset = adsClient.CreateVariableHandle(variableNames[i]);
            variables[i].length = variableLengths[i];
        }
        }
        catch (Exception err)
        {
        MessageBox.Show(err.Message);
        adsClient = null;
        }
    }
```

After clicking *'Read'* Button, *'BlockRead'* method returns an ADS Stream. Check for ADS return codes (Err) for error handling before data is read out of the stream und stored in the text boxes.

```
private void button1_Click(object sender, System.EventArgs e)
    {
        if (adsClient == null)
        return;

        try
        {
        // Get the ADS return codes and examine for errorsBinaryReader reader = newBinaryReader(Bloc
kRead(variables));
        for (int i = 0; i < variables.Length; i++)
        {
            int error = reader.ReadInt32();
            if (error != (int)AdsErrorCode.NoError)
            System.Diagnostics.Debug.WriteLine(
                String.Format("Unable to read variable {0} (Error = {1})", i, error));
        }

        // Read the data from the ADS streamMyStruct myStruct;
        myStruct.uintValue = reader.ReadUInt16();
        myStruct.dintValue = reader.ReadInt32();
        myStruct.boolValue = reader.ReadBoolean();

        // Write data from the structure into the text boxes
        tbUint.Text = myStruct.uintValue.ToString();
        tbDint.Text = myStruct.dintValue.ToString();
        tbBool.Text = myStruct.boolValue.ToString();
        }
        catch (Exception err)
        {
        MessageBox.Show(err.Message);
        }
    }
```

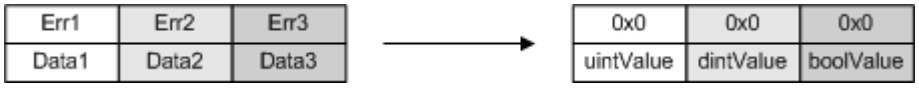
'BlockRead' methode takes an object of type *'VariableInfo[]'*, where the handle parameters are strored. After reserving memory for the values to be read and write, an ADS stream is written with the parameters delivered by *'VariableInfo[]'*. At the end the sum command transfers the commands, and an ADS stream is returned.

```
private AdsStream BlockRead(VariableInfo[] variables)
{
        // Allocate memoryint rdLength = variables.Length * 4;
        int wrLength = variables.Length * 12;

        // Write data for handles into the ADS StreamBinaryWriter writer = newBinaryWriter(newAdsStr
eam(wrLength));
        for (int i = 0; i < variables.Length; i++)
        {
        writer.Write(variables[i].indexGroup);
        writer.Write(variables[i].indexOffset);
        writer.Write(variables[i].length);
        rdLength += variables[i].length;
        }

        // Sum command to read variables from the PLCAdsStream rdStream = newAdsStream(rdLength);
        adsClient.ReadWrite(0xF080, variables.Length, rdStream, (AdsStream)writer.BaseStream);

        // Return the ADS error codesreturn rdStream;
    }
```

Sum Command's Parameters consists of *IndexGroup (0xF080)* - sum command call, *IndexOffset (variables.Length)* - number of sub commands, *rdDataStream (rdStream)* - memory, taking the read values, *wrDataStream (writer.BaseStream)* - memory, containing values to be written.

sub commands in *wrDataStream*

response in *rdDataStream*



2. writing variables After clicking *'Write'* button, *'BlockRead2'* methode returns an ADS stream. Following is a check of ADS return codes (Err) for error handlings.

```
private void button2_Click(object sender, EventArgs e)
{
        if (adsClient == null)
        return;

        try
        {
        // Get the ADS return codes and examine for errorsBinaryReader reader = newBinaryReader(Bloc
kRead2(variables));
        for (int i = 0; i < variables.Length; i++)
        {
            int error = reader.ReadInt32();
            if (error != (int)AdsErrorCode.NoError)
            System.Diagnostics.Debug.WriteLine(
                String.Format("Unable to read variable {0} (Error = {1})", i, error));
        }
        }
        catch (Exception err)
        {
        MessageBox.Show(err.Message);
        }
}
```

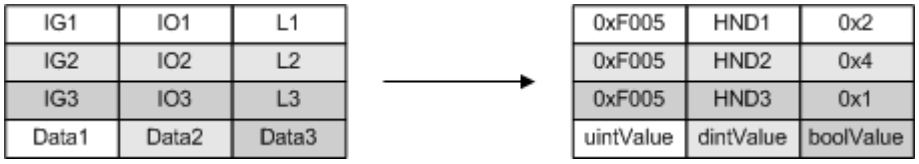
'BlockRead2' methode takes an object of type *'VariableInfo[]'*, where handle parameters are stored. After reserving memory for the values to be read an write, *'MyStruct'* object is filled with the values stored in the textboxes. Next, an ADS stream containing parameters from *'VariableInfo[]'* objects an the values of *'MyStruct'* object is written. At the end sum command transfers the commands and an ADS Stream is returned.

```
Private AdsStream BlockRead2(VariableInfo[] variables)
{
        // Allocate memoryint rdLength = variables.Length * 4;
        int wrLength = variables.Length * 12 + 7;

        BinaryWriter writer = newBinaryWriter(new AdsStream(wrLength));
        MyStruct myStruct;
        myStruct.uintValue = ushort.Parse(tbUint2.Text);
        myStruct.dintValue = int.Parse(tbDint2.Text);
        myStruct.boolValue = bool.Parse(tbBool2.Text);

        // Write data for handles into the ADS streamfor (int i = 0; i < variables.Length; i++)
        {
        writer.Write(variables[i].indexGroup);
        writer.Write(variables[i].indexOffset);
        writer.Write(variables[i].length);
        }

        // Write data to send to PLC behind the structure
```

```
        writer.Write(myStruct.uintValue);
        writer.Write(myStruct.dintValue);
        writer.Write(myStruct.boolValue);

        // Sum command to write the data into the PLCAdsStream rdStream = newAdsStream(rdLength);
        adsClient.ReadWrite(0xF081, variables.Length, rdStream, (AdsStream)writer.BaseStream);

        // Return the ADS error codesreturn rdStream;
}
```

Sum Command's Parameters consists of *IndexGroup (0xF081)* - sum command call, *IndexOffset (variables.Length)* - number of sub commands, *rdDataStream (rdStream)* - memory, taking the read values, *wrDataStream (writer.BaseStream)* - memory, containing values to be written.

sub commands in *wrDataStream*



response in *rdDataStream*

6.2.10.1 ADS sum command: reading or writing several variables with Delphi

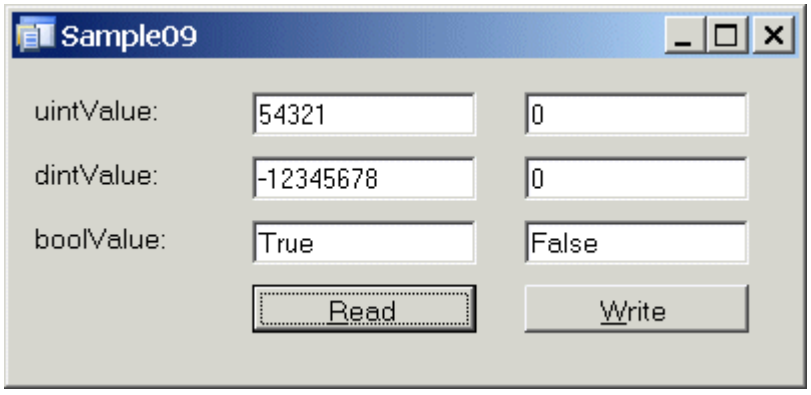
System requirements:

- **TwinCAT v2.11 Build >= 1550**;

Task

Read or write several PLC variables with TcAdsClient.ReadWrite command. This sample shows also how to get and enable several PLC variables handles with a ADS sum command.

Description



The singles sub commands are moved in an AdsStream (Write Data) to the PLC and executed there in a row. The result is moved with a further AdsStream (Return value + Read Data) from the PLC to the client.

Dynamic arrays can be used to assign a variable number of parameters (ADS sub commands). The Sample uses the dynamic arrays and makes the code universal reusable. Other methods are possible (array with fixed size, structure elements, etc.)

The following auxiliary methods are used as generic ADS sum commands to write or read the PLC variables value, to get or enable handles, etc. In these methods the AdsStream for the ADS sum command is filled with parameters and the ADS sum command is called by the TcAdsClient.ReadWrite command.

```
method BlockRead(
  ig : Array of Int32;
  io : Array of Int32;
  var rLen : Array of Int32;
  var data : AdsStream
) : Array of Int32;

method BlockWrite(
  ig : Array of Int32;
  io : Array of Int32;
  wLen : Array of Int32;
  data : AdsStream
) : Array of Int32;

method BlockReadWrite(
  ig, io, wLen : Array of Int32;
  wData : AdsStream;
  var rLen : Array of Int32;
  var rData : AdsStream
) : Array of Int32;
```

The generic methods are used by four more methods:

```
method BlockCreateVariableHandle(names : Array Of String) : Array of Int32;
method BlockDeleteVariableHandle(handles : Array of Int32) : Array of Int32;
```

```
method btnRead_Click(sender: System.Object; e: System.EventArgs);
method btnWrite_Click(sender: System.Object; e: System.EventArgs);
```

Delphi Prism (Embarcadero Prism XE2, Oxygen for .NET) program

```
namespace Sample09;

interface

uses
  System.Drawing,
  System.Collections,
  System.Collections.Generic,
  System.Windows.Forms,
  System.ComponentModel, System.IO, TwinCAT.Ads;

type
  /// <summary>
  /// Summary description for MainForm.
  /// </summary>
  MainForm = partial class(System.Windows.Forms.Form)
  private
    adsClient : TcAdsClient;
    vNames  : Array of String := ["MAIN.uintValue", "MAIN.dintValue", "MAIN.boolValue"];
    vLengths : Array of Int32 := [2,4,1];
    vHandles : Array of Int32 := [0,0,0];

    method BlockRead(ig : Array of Int32; io : Array of Int32; var rLen : Array of Int32; var data :
 AdsStream) : Array of Int32;
    method BlockWrite(ig : Array of Int32; io : Array of Int32; wLen : Array of Int32; data : AdsStr
eam) : Array of Int32;
    method BlockReadWrite(ig, io, wLen : Array of Int32;  wData : AdsStream; var rLen : Array of Int
32; var rData : AdsStream) : Array of Int32;

    method BlockCreateVariableHandle(names : Array Of String) : Array of Int32;
    method BlockDeleteVariableHandle(handles : Array of Int32) : Array of Int32;

    method MainForm_Load(sender: System.Object; e: System.EventArgs);
    method MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEventArgs)
;
    method btnRead_Click(sender: System.Object; e: System.EventArgs);
    method btnWrite_Click(sender: System.Object; e: System.EventArgs);
  protected
```

```
    method Dispose(disposing: Boolean); override;
  public
    const ADSIGRP_SUMUP_READ = $0000F080;       // AdsRW  IOffs list size or 0 (=0 -
> list size == WLength/3*sizeof(ULONG))                  // W: {list of IGrp, IOffs, Length}
                    // if IOffs != 0 then R: {list of results} and {list of data}
                    // if IOffs == 0 then R: only data (sum result)

    const ADSIGRP_SUMUP_WRITE = $0000F081;      // AdsRW  IOffs list size                  // W: {li
st of IGrp, IOffs, Length} followed by {list of data}
                    // R: {list of results}

    const ADSIGRP_SUMUP_READWRITE = $0000F082;// AdsRW  IOffs list size                  // W: {list
 of IGrp, IOffs, RLength, WLength} followed by {list of data}
                    // R: {list of results, RLength} followed by {list of data}
    constructor;
  end;

implementation

{$REGION Construction and Disposition}
constructor MainForm;
begin
  //
  // Required for Windows Form Designer support
  //
  InitializeComponent();
  //
  // TODO: Add any constructor code after InitializeComponent call
  //
end;

method MainForm.Dispose(disposing: Boolean);
begin
  if disposing then begin
    if assigned(components) then
      components.Dispose();
    //
    // TODO: Add custom disposition code here
    //
  end;
  inherited Dispose(disposing);
end;
{$ENDREGION}
```

Create connection

On starting the application a connection to the PLC is established and variable handles are fetched via theBlockCreateVariableHandlemethod call. The names of the three PLC variables are transferred to theBlockCreateVariableHandlemethod as a dynamic array.

```
//////////////////////////////////////////////////////////////////////////////////////////////////
//////
// Create form, connection and variable handles
method MainForm.MainForm_Load(sender: System.Object; e: System.EventArgs);
begin
  try
    // Connect to PLC
    adsClient := new TcAdsClient();
    adsClient.Connect(801);

    // Create variable handles
    vHandles := BlockCreateVariableHandle(vNames);

//    Conventional method:
//    for each name in vNames index i do
//      vHandles[i] := adsClient.CreateVariableHandle(name);

  except
    On err : AdsErrorException do
      MessageBox.Show(err.Message, err.Source);
  end;
end;
```

Close connection

On terminating the application the variable handles are released and the connection to the PLC is cut. The handles to be released are transferred to the BlockDeleteVariableHandle method as a dynamic array.

```
////////////////////////////////////////////////////////////////////////////////////////////////////
/////
// Close form and release resources
method MainForm.MainForm_FormClosing(sender: System.Object; e: System.Windows.Forms.FormClosingEvent
Args);
begin
  try
    // Release variable handles
    BlockDeleteVariableHandle(vHandles);

//  Conventional method
//    for each handle in vHandles index i do
//      adsClient.DeleteVariableHandle(handle);
    // Close connection
    adsClient.Dispose()
  except
    On err : AdsErrorException do
      MessageBox.Show(err.Message, err.Source)
  end;
end;
```

Read PLC values

The PLC variable values are read by clicking on theReadbutton. ThebtnRead_Click event handling routing uses the generic *BlockRead* method. However, the instance of the AdsStream for the PLC data must be generated beforehand with a suitable size so that the values of all PLC variables fit into it. The parameters rLen(read lengths)and data (AdsStream) are transferred by reference. As a result, theBlockRead method can fill the AdsStream with data and change the actually returned data lengths (rLen) (e.g.: in case of error). If successful, the data are successively read from the AdsStream and displayed on the form.

```
////////////////////////////////////////////////////////////////////////////////////////////////////
//////
// Read block data (multiple PLC values)
method MainForm.btnRead_Click(sender: System.Object; e: System.EventArgs);
begin
  var ig := new Int32[vHandles.Length];
  // Calculate the size of PLC value data
  var cbRead : Int32 := 0;
  for each len in vLengths index i do
  begin
    cbRead := cbRead + len;
    ig[i] := Int32(AdsReservedIndexGroups.SymbolValueByHandle);
  end;

  var data := new AdsStream(cbRead);
  var reader := new BinaryReader(data);
  var rLen := new Int32[vLengths.Length];
  rLen := vLengths;

  // Read PLC values
var blockResult := BlockRead(ig, vHandles, var rLen, var data);
  for each adsResult in blockResult index i do
    if ( adsResult <> 0 ) then
      MessageBox.Show("Read failed! Error: " + adsResult.ToString() , "TwinCAT.Ads");

  data.Position := 0;
  tbRuint.Text := iif((blockResult[0] = 0) And (rLen[0] = 2), reader.ReadUInt16().ToString(), "0");
  tbRdint.Text := iif((blockResult[1] = 0) And (rLen[1] = 4), reader.ReadInt32().ToString(), "0");
  tbRbool.Text := iif((blockResult[2] = 0) And (rLen[2] = 1), reader.ReadBoolean().ToString(), "Fals
e");
end;
```

Write PLC values

The values are written from the application to the PLC by clicking on the *Write* button. The *btnWrite_Click* event handling routine uses the generic *BlockWrite* method. The instance of the AdsStream is generated before the call with a suitable size and filled with the PLC variable values to be written.

```
////////////////////////////////////////////////////////////////////////////////////////////////////
//////
// Write block data (multiple PLC values)
method MainForm.btnWrite_Click(sender: System.Object; e: System.EventArgs);
begin
  var ig := new Int32[vHandles.Length];
  // Calculate the size of PLC value data
  var cbWrite : Int32 := 0;
  for each len in vLengths index i do
  begin
    cbWrite := cbWrite + len;
    ig[i] := Int32(AdsReservedIndexGroups.SymbolValueByHandle);
  end;

  var data := new AdsStream(cbWrite);
  var writer := new BinaryWriter(data);

  writer.Write(UInt16.Parse(tbWuint.Text));
  writer.Write(Int32.Parse(tbWdint.Text));
  writer.Write(boolean.Parse(tbWbool.Text));

  // Write PLC values
 var blockResult := BlockWrite(ig, vHandles, vLengths, data);
  for each adsResult in blockResult do
    if adsResult <> 0 then
      MessageBox.Show("Write failed! Error: " + adsResult.ToString() , "TwinCAT.Ads");
end;
```

Create variable handles

The generic *BlockReadWrite* method is used to fetch the variable handles. The symbol names of the PLC variables are transferred to the *BlockReadWrite* method as write data in the AdsStream. The handles are received in a further AdsStream as read data. As with the *BlockWrite* method, the AdsStream: *rData* (read data) and *rLen* are transferred by reference here also. Only in this way can the contents of the stream be filled with data from the *BlockReadWrite* method. If successful, the handles are returned by the method as return parameters in the form of a dynamic array.

```
////////////////////////////////////////////////////////////////////////////////////////////////////
//////
// Create multiple symbol handles
method MainForm.BlockCreateVariableHandle( names : Array Of String ) : Array of Int32;
begin
  var getHandles := new Int32[names.Length];
  var ig := new Int32[names.Length];
  var io := new Int32[names.Length];
  var rLen := new Int32[names.Length];
  var wLen := new Int32[names.Length];
  var cbWrite : Int32 := 0;
  for each name in names do
    cbWrite := cbWrite + name.Length;
  var wData := new AdsStream(cbWrite);
  var writer := new BinaryWriter(wData, System.Text.Encoding.ASCII);

  var rData := new AdsStream(getHandles.length * 4);
  var reader := new BinaryReader(rData);

  for each name in names index i do
  begin
    writer.Write(name.ToCharArray());// Copy PLC variable names to the stream
    ig[i] := Int32(AdsReservedIndexGroups.SymbolHandleByName);// indexGroup
    io[i] := 0;// indexOffset = 0
    rLen[i] := sizeOf(getHandles[i]);// readLen = 4 byte
    wLen[i] := name.Length;// writeLen = length of plc variable name
  end;

  var blockResult := BlockReadWrite(ig, io, wLen, wData, var rLen, var rData);
  rData.Position := 0;
  for each adsResult in blockResult index i do
  begin
```

```
    if (adsResult = 0) and (rLen[i] = 4) then
      getHandles[i] := reader.ReadInt32()
    else
      MessageBox.Show("CreateVariableHandle failed! Error: " + adsResult.ToString() , "TwinCAT.Ads")
;
  end;

  result := getHandles;
end;
```

Realease variable handles

The method *BlockDeleteVariableHandle* uses the method *BlockWrite*. The handles to be enabled are copied before in the AdsStream (Write-Data).

```
////////////////////////////////////////////////////////////////////////////////////////////////////
//////
// Release multiple symbol handles
method MainForm.BlockDeleteVariableHandle(handles : Array of Int32 ): Array of Int32;
begin
  var data := new AdsStream(handles.Length * 4);
  var writer := new BinaryWriter(data);
  var ig := new Int32[handles.Length];
  var wLen := new Int32[handles.Length];

  for each handle in handles index i do
  begin
    writer.Write(handle);
    ig[i] := Int32(AdsReservedIndexGroups.SymbolReleaseHandle);
    wLen[i] := sizeOf(handle);// = 4 byte
  end;

  var blockResult := BlockWrite(ig, handles, wLen, data);
  for each adsResult in blockResult do
    if adsResult <> 0 then
      MessageBox.Show("DeleteVariableHandle failed! Error: " + adsResult.ToString() , "TwinCAT.Ads")
;

  result := blockResult;
end;
```

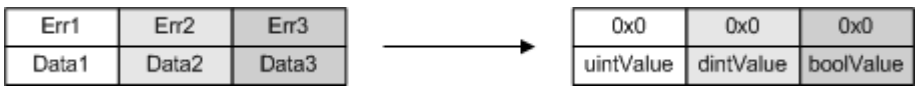
Generic ADS read sum command

The parameter for the ADS sum command consists of:

- IndexGroup =ADSIGRP_SUMUP_READ= *0xF080;*
- IndexOffset =Number of ADS sub commands (in the sample = 3);
- readData (AdsStream) = Memory that takes read data = ListOf(Ads result) + ListOf(data);
- writeData(AdsStream)=Memory that contains data to be sent = ADS sub commands = ListOf(indexGroup + indexOffset + readLength);

Arrangement of the ADS sub command inwriteDataduring writing the PLC variables values:

Arrangement of return data inreadDataduring reading the PLC variables values:



The return values of the single ADS subcommands are extracted from thereadData-Stream and returned fromtheBlockWritemethod as return value. The read data (e.g. PLC variables values) are copied into the data method parameter.

The rLen values are aligned (in error case the data length = Null).

```
////////////////////////////////////////////////////////////////////////////////////////////////////
//////
// Ads block read
method MainForm.BlockRead( ig : Array of Int32; io : Array of Int32; var rLen : Array of Int32;  var
 data : AdsStream) : Array of Int32;
begin
  var count := ig.Length;
  var blockResult := new Int32[count];
  try
    var writeData := new AdsStream(count * 12);
    var writer := new BinaryWriter(writeData);
    var cbData : Int32 := 0;

    for each len in rLen index i do
    begin
      writer.Write(ig[i]);
      writer.Write(io[i]);
      writer.Write(len);
      cbData := cbData + len;
    end;

    if cbData <> data.Length then
      MessageBox.Show("BlockRead()::Invalid read data length!", "TwinCAT.Ads");

    var readData := new AdsStream((count * 4) + data.Length);
    var reader := new BinaryReader(readData);

    adsClient.ReadWrite(ADSIGRP_SUMUP_READ, count, readData, writeData);

    readData.Position := 0;
    for each adsResult in blockResult index i do
      blockResult[i] := reader.ReadInt32();

    // Copy read data
    var copy := new BinaryWriter(data);
    data.Position := 0;
    for each adsResult in blockResult index i do
    begin
      if adsResult = 0 then// no error => copy data, error => don't copy data
    copy.Write(reader.ReadBytes(rLen[i]))
      else
    rLen[i] := 0;
    end;
  except
    On err : AdsErrorException do
      MessageBox.Show(err.Message, err.Source)
  end;

  result := blockResult;
end;
```

Generic ADS write sum command

The parameter of the ADS write sum command consists of:

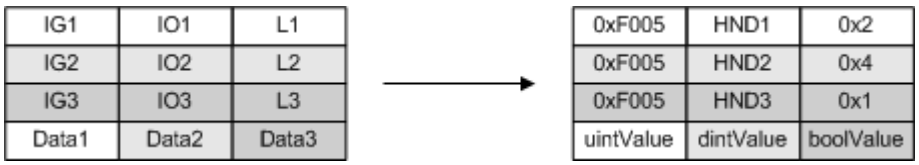
- IndexGroup =ADSIGRP_SUMUP_WRITE= *0xF081;*
- IndexOffset = Number of ADS sub commands(in the sample = 3);
- readData (AdsStream) = Memory that takes read data = ListOf(Ads result);
- writeData (AdsStream) =Memory that contains data to be sent = ADS sub commands = ListOf(indexGroup + indexOffset + writeLength) + ListOf(data);

TheBlockWrite method is used in two cases (in this sample):

- Writing the PLC variables values => the method parameterdatacontains the new PLC variables values;
- Enable the variable handles => the method parameterdatacontains handles to be enabled;

Thecontent of data is copied into writeData.

Arrangement of the ADS sub command inwriteDataduring writing the PLC variables values:



Arrangement of return data inreadDataduring reading the PLC variables values:

The return values of the single ADS subcommands are extracted from thereadData-Stream and returned fromtheBlockWritemethod as return value.

```
////////////////////////////////////////////////////////////////////////////////////////////////////
//////
// Ads block write
method MainForm.BlockWrite(ig : Array of Int32; io : Array of Int32; wLen : Array of Int32; data : A
dsStream) : Array of Int32;
begin
  var count := ig.Length;
  var blockResult := new Int32[count];
  try
    var writeData := new AdsStream((count * 12 ) + data.Length);
    var writer := new BinaryWriter(writeData);
    var cbData : Int32 := 0;

    for each len in wLen index i do
    begin
      writer.Write(ig[i]);
      writer.Write(io[i]);
      writer.Write(len);
      cbData := cbData + len;
    end;

    if cbData <> data.Length then
      MessageBox.Show("BlockWrite()::Invalid write data length!", "TwinCAT.Ads");

    writer.Write(data.ToArray());

    var readData := new AdsStream(count * 4);
    var reader := new BinaryReader(readData);

    adsClient.ReadWrite(ADSIGRP_SUMUP_WRITE, count, readData, writeData);
    readData.Position := 0;
    for each adsResult in blockResult index i do
      blockResult[i] := reader.ReadInt32();

  except
    On err : AdsErrorException do
      MessageBox.Show(err.Message, err.Source)
  end;

  result := blockResult;
end;
```

Generic ADS read/write sum command

The BlockReadWrite method is a combination ofBlockRead andBlockWritemethod. The parameter of the ADS readwrite sum command consists of:

- IndexGroup =ADSIGRP_SUMUP_READWRITE= *0xF082;*
- IndexOffset = Number of ADS sub commands(in the sample = 3);
- readData(AdsStream) = Memory that takes read data = ListOf(Ads result + readLength) + ListOf(data);
- writeData (AdsStream)=Memory that contains data to be sent = ADS sub commands = ListOf(indexGroup + indexOffset + readLength + writeLength) + ListOf(data);

The *BlockReadWrite* method is used in this example when fetching the handles. In this case the *wData* method parameter contains the strings with the PLC symbol names. The symbol names are copied into *writeData* before the call. The handles are located in *readData* after the call and are copied from there into the *rData* method parameter. The *rLen* values are also adapted (in case of error the data length = zero). The return values of the individual ADS sub-commands are extracted from the *readData* stream and returned by the *BlockReadWrite* method as a return value.

```
////////////////////////////////////////////////////////////////////////////////////////////////////
//////
// Ads block read/write
method MainForm.BlockReadWrite( ig, io, wLen : Array of Int32;  wData : AdsStream;
                  var rLen : Array of Int32; var rData : AdsStream) : Array of Int32;
begin
  var count := ig.Length;
  var blockResult := new Int32[count];
  try
    var writeData := new AdsStream((count * 16) + wData.Length);// 16 = sizeof(indexGroup + indexOff
set + readLen + writeLen)
    var writer := new BinaryWriter(writeData);
    var cbWData : Int32 := 0;
    var cbRData : Int32 := 0;

    for each len in wLen index i do
    begin
      writer.Write(ig[i]);
      writer.Write(io[i]);
      writer.Write(rLen[i]);
      writer.Write(wLen[i]);
      cbWData := cbWData + wLen[i];
      cbRData := cbRData + rLen[i];
    end;

    if cbRData <> rData.Length then
      MessageBox.Show("BlockReadWrite()::Invalid read data length!", "TwinCAT.Ads");

    if cbWData <> wData.Length then
      MessageBox.Show("BlockReadWrite()::Invalid write data length!", "TwinCAT.Ads");

    writer.Write(wData.ToArray());

    var readData := new AdsStream((count * 8) + rData.Length);
    var reader := new BinaryReader(readData);

    adsClient.ReadWrite(ADSIGRP_SUMUP_READWRITE, count, readData, writeData);

    readData.Position := 0;
    for each adsResult in blockResult index i do
    begin
      blockResult[i] := reader.ReadInt32();
      rLen[i] := reader.ReadInt32();
    end;

    // Copy data
    var copy := new BinaryWriter(rData);
    rData.Position := 0;
    for each adsResult in blockResult index i do
    begin
      if adsResult = 0 then// no error => copy data, error => don't copy data
    copy.Write(reader.ReadBytes(rLen[i]));
    end;

  except
    On err : AdsErrorException do
      MessageBox.Show(err.Message, err.Source)
```

```
  end;

  result := blockResult;
end;

end.
```

PLC program

```
PROGRAM MAIN
VAR
    uintValue : UDINT := 54321;
    boolValue : BOOL := TRUE;
    dintValue : DINT := -12345678;
END_VAR
```

```
;
```

Download

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygen for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466775819/.exe

6.2.11 Reading of SMB values from TwinCAT I/O driver

Download

Language / IDE	Extract the sample program
Visual C#	-
Visual Basic (for .NET Framework)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490159755/.exe
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	-
Delphi for .NET (Borland Developer Studio 2006)	-

TX1000 | TwinCAT CP (royalty-free) is a product containing drivers for the Beckhoff Control Panels respectively Control Panel-PC's of the CP6xxx und CP7xxx series, the industrial human machine interfaces from Beckhoff.
It gets typically used if there is no full TwinCAT version considered but basic functions of the Control Panel or Control Panel-PC need to get accessed via application software (like pushbutton extensions or the named SMB (System Management Bus) on supported Motherboards, etc...
TwinCAT CP is a subset of the full TwinCAT suite, means all contained drivers are binary identical to those in the other TwinCAT levels available. Written application software doesn't need to get touched in any way, if TwinCAT CP gets upgraded to a full TwinCAT version later.

Even if only TwinCAT-CP is installed on the controller or HMI device, the SMB device can get accessed by the ADS client (i.e. from a Visual Basic or similar DLL-, Active X- or .NET-capable application).

The visualization allows a continuous supervision of operation temperatures, fan speeds or even the different power supply voltages. That way, the crossing of a critical threshold value can be detected (i.e. if an implementation of an advanced maintenance system is considered). The following example shows a possible implementation under Visual Basic.NET.

Order options:

TwinCAT-CP | License for using the I/O driver for Beckhoff Control Panels for communication between generic Windows applications (alternatively, any other level of TwinCAT of course)

Required software:

- Microsoft.NET Framework version 2.0: http://msdn2.microsoft.com
- Microsoft Visual Studio 2005 or 2008: http://msdn2.microsoft.com
  - to create the application "Motherboard Diagnosis“ (see below)
- TwinCAT 2.10: https://www.beckhoff.com/en-en/products/automation/twincat/txxxxx-twincat-2-base/

  - during a new installation of TwinCAT:
    To install the royalty-free TwinCAT "CP" level, it's important that TwinCAT level "CP" and no other optional level gets selected during the initial installation.

    :*Note* During an Update installation, the previously selected level cannot be changed. Instead, TwinCAT needs to get uninstalled first (in case of a wished upgrade/downgrade).

Visual Basic (for .NET Framework) program

Example: Temperature

1. Add Reference

At first, a *Reference* to the necessary TwinCAT.Ads.dll needs to be added. The TwinCAT.ADS.NET component gets added via right mouse-click on *Project→Add Reference*.

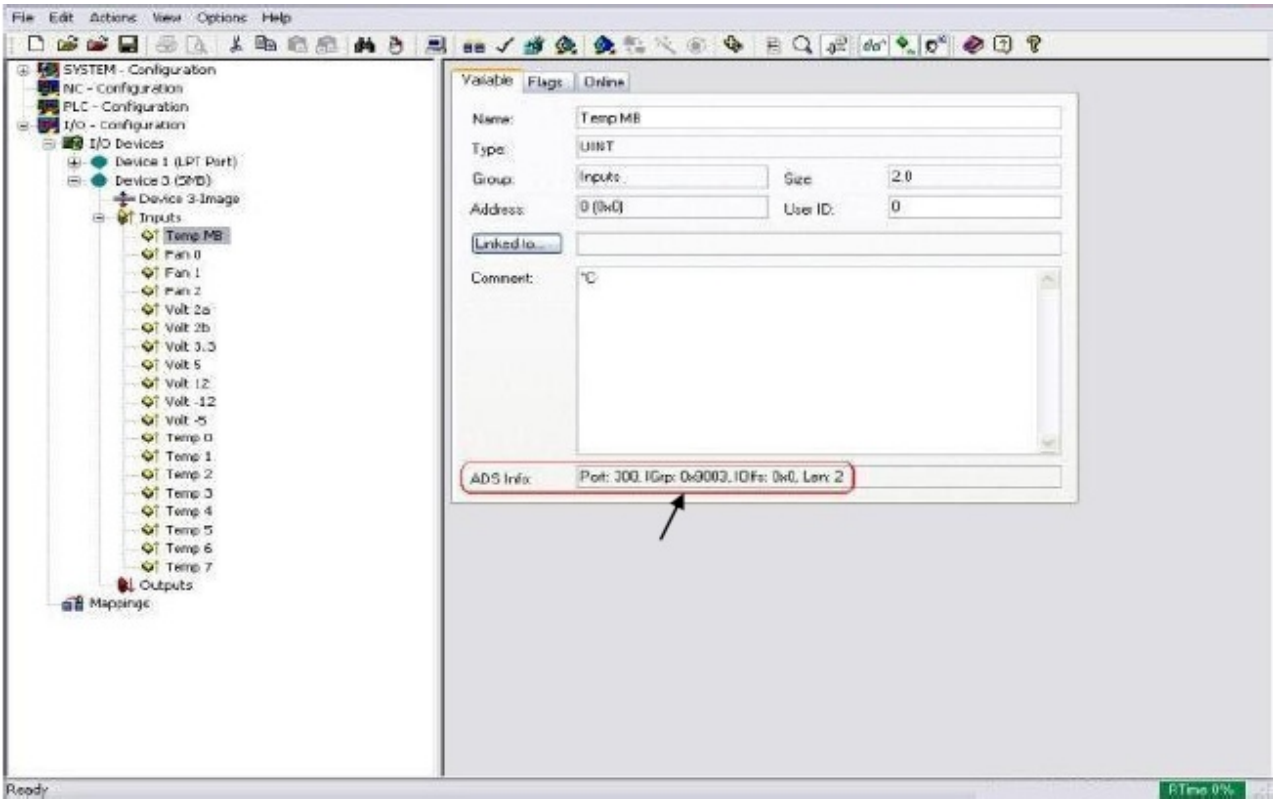
2. Determination of device address (here) *: Index-Group*

```
Private Sub MotherboardDiagnosis_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    Try
        'Create a new instance of class TcAdsClient
                    tcClient = New TwinCAT.Ads.TcAdsClient()

        'Connect to local PLC - Runtime 1 - Port 300
        tcClient.Connect(300)

        'Gets the device count of devices appended to active configuration if
        'TwinCAT is in run mode
        Dim state = tcClient.ReadState.AdsState

        If Not state = AdsState.Run Then
            txtCurrent.Text = "Bad TwinCAT state ("& state.ToString() &").
            Please restart application."
        Else
            Dim deviceCount As UInt32
            deviceCount = tcClient.ReadAny(&H5000, &H2,
            deviceCount.GetType())


            'Gets the device count of devices appended to active
            'configuration as zero-based index
            'and gets the device ID's of all active devices for the
            'remaining indices (inverted sort)
            Dim arrayDeviceIDs(deviceCount) As UInt16
            arrayDeviceIDs = tcClient.ReadAny(&H5000, &H1,
            arrayDeviceIDs.GetType(), New Integer() {deviceCount + 1})
            For i As Integer = 1 To (deviceCount)
                'gets the device identification number (Motherboard
                'System Management Bus(SMB) => 32)
                Dim devIdent = tcClient.ReadAny(arrayDeviceIDs(i) +
                &H5000, &H7, arrayDeviceIDs(i).GetType)


                'checks if the device type is a SMB-Type
                If devIdent = 32 Then
                    _iIndexGroup = &H9000 + arrayDeviceIDs(i)
            Exit For
        End If

        Next
```

```
        'Enable timer to start monitoring with determined IGroup
        tmrTimer1.Enabled = True
    End If

    Catch ex As Exception
        txtCurrent.Text = ex.Message
    End Try
End Sub
```

3. Determination of the Variable address (here:) *Index-Offset*

To determine the *Index-Offset* of the variable, it's very comfortable to use TwinCAT System Manager . By selecting the required *Input* in the left tree-view, the related window on the right shows the whished address in hexa-decimal notation.

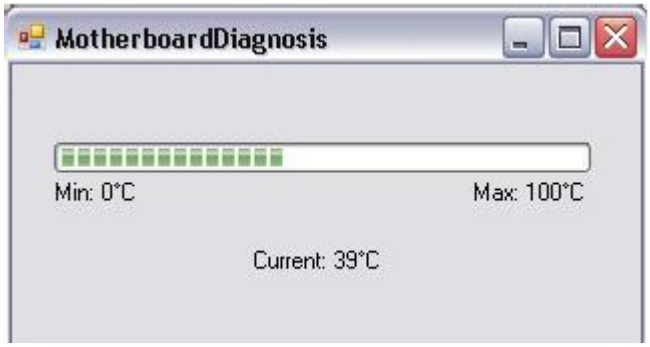


4. Cyclic read update of the value in your application

```
Private Sub tmrTimer1_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles tmrTimer1.Tick
    Try
        'IOffset may be adjusted to monitor different inputs e.g. fan-speeds
        'or voltages
        _iTemp = tcClient.ReadAny(_iIndexGroup, 0, _iTemp.GetType())
        ProgressBar1.Value = _iTemp
        lblCurrent.Text = ("Current: " + (Convert.ToString(_iTemp)) + "°C")

        Catch ex As Exception
            'Catches if certain circumstances turned TwinCAT to offline mode.
            'Restarts monitoring if configuration remains unchanged and TwinCat
            'is turned online again.
            If ex.Message = "Ads-Error 0x12 : Port is disabled." Then
                lblCurrent.Text = "Ports are disabled. TwinCAT may be in
                offline mode."
                ProgressBar1.Value = 0
            Else
                MessageBox.Show(ex.Message)
```

```
            End If
    End Try
End Sub
```

Example: Resulting Message box when exceeding a temperature threshold value



6.2.12 Delete a handle of a PLC variable

Download

Language / IDE	Extract the sample program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490161163/.zip
Visual Basic (for .NET Framework)	-
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490162571/.exe
Delphi for .NET (Borland Developer Studio 2006)	-

Description

This Sample shows how to delete a handle of a PLC variable:

C# program

```
static void Main(string[] args)
{
    //Create a new instance of class TcAdsClient
    TcAdsClient tcClient = new TcAdsClient();
    int iHandle = 0;

    try
    {
    // Connect to local PLC - Runtime 1 - TwinCAT2 Port=801, TwinCAT3 Port=851
    tcClient.Connect(801);

        //Get the handle of the PLC variable "PLCVar"
        iHandle = tcClient.CreateVariableHandle("MAIN.PLCVar");

        //Release the specific handle of "PLCVar"
        tcClient.DeleteVariableHandle(iHandle);
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
        Console.ReadKey();
    }
    finally
    {
        tcClient.Dispose();
    }
}
```

6.2.12.1 Delete a handle of a PLC variable with Delphi

Console application. This Sample shows how to delete a handle of a PLC variable.

Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample11;

interface
uses TwinCAT.Ads;

type
  ConsoleApp = class
  public
    class method Main(args: array of string);
  end;

implementation

class method ConsoleApp.Main(args: array of string);
begin
  // Create a new instance of class TcAdsClient
  var tcClient := new TcAdsClient();
  var iHandle : Int32;

  try
    // Connect to local PLC - Runtime 1 - Port 801
    tcClient.Connect(801);

    // Get the handle of the PLC variable "MAIN.PLCVar"
    iHandle := tcClient.CreateVariableHandle("MAIN.PLCVar");

    // Use the handle to read from PLC variable
    var sOldValue := tcClient.ReadAny(iHandle, TypeOf(UInt32)).ToString();
    Console.WriteLine("PLCVar: " + sOldValue);

    // Use the handle to write to PLC variable, e.g. reset PLCVar
    var newValue := UInt32(0);
    tcClient.WriteAny(iHandle, newValue);
  except
    on err : Exception do
      Console.WriteLine(err.Message);
  finally
    // Release the variable handle
    tcClient.DeleteVariableHandle(iHandle);

    // Close connection to the PLC
    tcClient.Dispose();
  end;

  // Wait for user input
  Console.ReadKey();
end;

end.
```

PLC program

```
PROGRAM MAIN
VAR
    PLCVar : DWORD;
    timer : TON;
END_VAR
```

```
timer( IN := TRUE, PT := t#1s);
IF timer.Q THEN
    timer( IN := FALSE );
    timer( IN := TRUE );
    PLCVar := PLCVar + 1;
END_IF
```

Download

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466777227/.exe

6.2.13 Read flag synchronously from the PLC

Download

Language / IDE	Extract the sample program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490163979/.zip
Visual Basic (for .NET Framework)	-
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490165387/.exe
Delphi for .NET (Borland Developer Studio 2006)	-

Description

In this example program the value in flag double word 0 in the PLC is read and displayed on the screen:

C# program

```
static void Main(string[] args)
{
    //Create a new instance of class TcAdsClient
    TcAdsClient tcClient = new TcAdsClient();

    try
    {
    // Connect to local PLC - Runtime 1 - TwinCAT2 Port=801, TwinCAT3 Port=851
    tcClient.Connect(801);

     //Specify IndexGroup, IndexOffset and read SPSVar
     int iFlag = (int)tcClient.ReadAny(0x4020, 0x0, typeof(Int32));

        Console.WriteLine("" + iFlag);
     Console.ReadKey();
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
        Console.ReadKey();
    }
    finally
    {
        tcClient.Dispose();
    }
}
```

6.2.13.1 Read flag synchronously from the PLC with Delphi

Console application. In this example program the value in flag double word 0 in the PLC is read and displayed on the screen.

Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample12;

interface
uses System.IO,TwinCAT.Ads;

type
  ConsoleApp = class
  public
    class method Main(args: array of string);
  end;

implementation

class method ConsoleApp.Main(args: array of string);
begin
   Console.WriteLine("Please enter 'e' to exit or any key to read...");
  // Create a new instance of class TcAdsClient
  var tcClient := new TcAdsClient();
  try
    // Connect to local PLC - Runtime 1 - Port 801
    tcClient.Connect(801);
    // Read PLC variable
    while (Console.ReadKey().KeyChar <> 'e') do
    begin
      // Specify indexGroup, indexOffset and read PLCVar
      var dwFlag := tcClient.ReadAny($4020, $0, typeof(UInt32));
      Console.WriteLine("PLCVar: " + dwFlag.ToString());
    end;
  except
    on err : Exception do
    begin
      Console.WriteLine(err.Message);
      Console.ReadKey();
    end;
  finally
    // Close connection
    tcClient.Dispose();
  end;
end;

end.
```

PLC program

```
PROGRAM MAIN
VAR
    PLCVar AT %MD0 : DWORD;
    timer : TON;
END_VAR
```

```
timer( IN := TRUE, PT := t#1s);
IF timer.Q THEN
    timer( IN := FALSE );
    timer( IN := TRUE );
    PLCVar := PLCVar + 1;
END_IF
```

Download

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466778635/.exe

6.2.14 Write flag synchronously into the PLC

Download

Language / IDE	Extract the sample program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490166795/.zip
Visual Basic (for .NET Framework)	-
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490168203/.exe
Delphi for .NET (Borland Developer Studio 2006)	-

Description

In this example program, the value that the user has entered is written into flag double word 0:

C# program

```
static void Main(string[] args)
{
    //Create a new instance of class TcAdsClient
    TcAdsClient tcClient = new TcAdsClient();

    try
    {
    // Connect to local PLC - Runtime 1 - TwinCAT2 Port=801, TwinCAT3 Port=851
    tcClient.Connect(801);

        //Specify IndexGroup, IndexOffset and write SPSVar
        int iNewValue = 0;
    tcClient.WriteAny(0x4020, 0x0, iNewValue);
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
        Console.ReadKey();
    }
    finally
    {
    tcClient.Dispose();
    }
}
```

6.2.14.1 Write flag synchronously into the PLC with Delphi

Console application. In this example program, the value that the user has entered is written into flag double word 0.

Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample13;

interface
uses TwinCAT.Ads;

type
  ConsoleApp = class
  public
    class method Main(args: array of string);
  end;

implementation

class method ConsoleApp.Main(args: array of string);
begin
  Console.WriteLine("Please enter 'e' to exit or integer value and press Enter when ready...");
  // Create a new instance of class TcAdsClient
  var tcClient := new TcAdsClient();
  var sInput : String;
```

```
  var newValue : UInt32;
  try
    // Connect to local PLC - Runtime 1 - Port 801
    tcClient.Connect(801);
    repeat
      sInput := Console.ReadLine();
      if sInput = 'e' then
    break;
      newValue := UInt32.Parse(sInput);
      // Specify indexGroup, indexOffset and write PLCVar
      tcClient.WriteAny($4020, $0, newValue );
    until sInput = 'e';

  except
    on err : Exception do
    begin
      Console.WriteLine(err.Message);
      Console.ReadKey();
    end;
  finally
    // Close connection
    tcClient.Dispose();
  end;
end;

end.
```

PLC program

```
PROGRAM MAIN
VAR
    PLCVar AT %MD0 : DWORD;
END_VAR
```

```
;
```

Download

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466780043/.exe

6.2.15 Start/stop PLC

Download

Language / IDE	Extract the sample program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490169611/.zip
Visual Basic (for .NET Framework)	-
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490171019/.exe
Delphi for .NET (Borland Developer Studio 2006)	-

Description

The following program starts or stops run-time system 1 in the PLC:

C# program

```
static void Main(string[] args)
{
    //Create a new instance of class TcAdsClient
    TcAdsClient tcClient = new TcAdsClient();
```

```
    try
    {
    // Connect to local PLC - Runtime 1 - TwinCAT2 Port=801, TwinCAT3 Port=851
    tcClient.Connect(801);

        Console.WriteLine(" PLC Run\t[R]");
        Console.WriteLine(" PLC Stop\t[S]");
        Console.WriteLine("\r\nPlease choose \"Run\" or \"Stop\" and confirm with enter..");
        string sInput = Console.ReadLine().ToLower();

    //Process user input and apply chosen state
    do{
        switch (sInput)
        {
        case "r": tcClient.WriteControl(new StateInfo(AdsState.Run, tcClient.ReadState().DeviceState
)); break;
        case "s": tcClient.WriteControl(new StateInfo(AdsState.Stop, tcClient.ReadState().DeviceStat
e)); break;
        default: Console.WriteLine("Please choose \"Run\" or \"Stop\" and confirm with enter.."); sI
nput = Console.ReadLine().ToLower(); break;
        }
    } while (sInput != "r" && sInput != "s");
    }
    catch (Exception ex)
    {
    Console.WriteLine(ex.Message);
    Console.ReadKey();
    }
    finally
    {
    tcClient.Dispose();
    }
}
```

6.2.15.1 Start/stop PLC with Delphi

Console application. The following program starts or stops run-time system 1 in the PLC.

Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample14;

interface

uses System.IO, TwinCAT.Ads;

type
  ConsoleApp = class
  public
    class method Main(args: array of string);
  end;

implementation

class method ConsoleApp.Main(args: array of string);
begin
  Console.WriteLine("Please enter 'r' to RUN or 's' to STOP the PLC or any key to exit...");
  // Create a new instance of class TcAdsClient
  var tcClient := new TcAdsClient();
  var sInput : String;
  try
    // Connect to local PLC - Runtime 1 - Port 801
    tcClient.Connect(801);
    repeat
      // Get current state
      var oldState := tcClient.ReadState();
      Console.WriteLine("State: " + oldState.AdsState.ToString());

      // Read user input
      sInput := Console.ReadKey().KeyChar;
      if (sInput <> 'r') And (sInput <> 's') then
    break;

      // Set new state
      var newState := new StateInfo(iif(sInput = 'r', AdsState.Run, AdsState.Stop), oldState.DeviceS
tate);
```

```
      tcClient.WriteControl(newState);

    until (sInput <> 'r') And (sInput <> 's');
  except
    On err : Exception do
    begin
      Console.WriteLine(err.Message);
      Console.ReadKey();
    end;
  finally
    // Close connection
    tcClient.Dispose();
  end;

end;

end.
```

Download

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466781451/.exe

6.2.16 Access by variable name

Download

Language / IDE	Extract the sample program
Visual C#	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490172427/.zip
Visual Basic (for .NET Framework)	-
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12490173835/.exe
Delphi for .NET (Borland Developer Studio 2006)	-

Description

The following program accesses a PLC variable that does not have an address. Access must therefore be made by the variable name. Once the PLC variable in the example program exceeds 10 it is reset to 0:

C# program

```
static void Main(string[] args)
{
    //Create a new instance of class TcAdsClient
    TcAdsClient tcClient = new TcAdsClient();
    AdsStream dataStream = new AdsStream(4);
    AdsBinaryReader binReader = new AdsBinaryReader(dataStream);

    int iHandle = 0;
    int iValue = 0;

    try
    {
    // Connect to local PLC - Runtime 1 - TwinCAT2 Port=801, TwinCAT3 Port=851
    tcClient.Connect(801);

    //Get the handle of the PLC variable "PLCVar"
    iHandle = tcClient.CreateVariableHandle("MAIN.PLCVar");

    Console.WriteLine("Press enter to continue and any other key to abort..");

    do
    {
        //Use the handle to read PLCVar
```

```
        tcClient.Read(iHandle, dataStream);
        iValue = binReader.ReadInt32();
        dataStream.Position = 0;

        Console.WriteLine("Current value is: " + iValue);

        if (iValue >= 10)
        {
        //Reset PLC variable to zero
        tcClient.WriteAny(iHandle, 0);
        }

    } while (Console.ReadKey().Key.Equals(ConsoleKey.Enter));
    }
    catch (Exception ex)
    {
    Console.WriteLine(ex.Message);
    Console.ReadKey();
    }
    finally
    {
    //Delete variable handle
    tcClient.DeleteVariableHandle(iHandle);
    tcClient.Dispose();
    }
}
```

6.2.16.1 Access by variable name with Delphi

Console application. The following program accesses a PLC variable that does not have an address. Access must therefore be made by the variable name. Once the PLC variable in the example program exceeds 10 it is reset to 0.

Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET) program

```
namespace Sample15;

interface

uses System.IO, TwinCAt.Ads, System.Data, System.Runtime.InteropServices;

type
  ConsoleApp = class
  public
    class method Main(args: array of string);
  end;

implementation

class method ConsoleApp.Main(args: array of string);
begin
  Console.WriteLine("Please enter 'e' to exit or any key to read...");

  // Create a new instance of class TcAdsClient
  var tcClient := new TcAdsClient();
  var dwValue : UInt32 := 0;
  var iHandle : Int32 := 0;
  try
    // Connect to local PLC - Runtime 1 - Port 801
    tcClient.Connect(801);
    // Get the handle of the PLC variable "SPSVar"
    iHandle := tcClient.CreateVariableHandle("MAIN.PLCVar");

    while (Console.ReadKey().KeyChar <> 'e') do
    begin
      dwValue := UInt32(tcClient.ReadAny(iHandle, TypeOf(UInt32)));
      Console.WriteLine("Value: " + dwValue.ToString());

      if dwValue > 10 then
    tcClient.WriteAny(iHandle, UInt32(0));
    end;

  except
    On err : Exception do
```

```
    begin
      Console.WriteLine(err.Message);
      Console.ReadKey();
    end;
  finally
    // Release variable handle
    tcClient.DeleteVariableHandle(iHandle);

    // Close connection
    tcClient.Dispose();
  end;

end;

end.
```

PLC program

```
PROGRAM MAIN
VAR
    PLCVar : DWORD;
    timer : TON;
END_VAR
```

```
timer( IN := TRUE, PT := T#1s );
IF timer.Q THEN
    timer( IN := FALSE );
    timer( IN := TRUE );
    PLCVar := PLCVar + 1;
END_IF
```

Download

Language / IDE	Unpack the sample program
Delphi Prism (Embarcadero Prism XE2, Oxygene for .NET)	https://infosys.beckhoff.com/content/1033/tcadsnetref/Resources/12466782859/.exe

7 ADS Return Codes

Grouping of error codes:
Global error codes: ADS Return Codes [▶ 2108]... (0x9811_0000 ...)
Router error codes: ADS Return Codes [▶ 2108]... (0x9811_0500 ...)
General ADS errors: ADS Return Codes [▶ 2109]... (0x9811_0700 ...)
RTime error codes: ADS Return Codes [▶ 2110]... (0x9811_1000 ...)

Global error codes

Hex	Dec	HRESULT	Name	Description
0x0	0	0x98110000	ERR_NOERROR	No error.
0x1	1	0x98110001	ERR_INTERNAL	Internal error.
0x2	2	0x98110002	ERR_NORTIME	No real time.
0x3	3	0x98110003	ERR_ALLOCLOCKEDMEM	Allocation locked – memory error.
0x4	4	0x98110004	ERR_INSERTMAILBOX	Mailbox full – the ADS message could not be sent. Reducing the number of ADS messages per cycle will help.
0x5	5	0x98110005	ERR_WRONGRECEIVEHMSG	Wrong HMSG.
0x6	6	0x98110006	ERR_TARGETPORTNOTFOUND	Target port not found – ADS server is not started or is not reachable.
0x7	7	0x98110007	ERR_TARGETMACHINENOTFOUND	Target computer not found – AMS route was not found.
0x8	8	0x98110008	ERR_UNKNOWNCMDID	Unknown command ID.
0x9	9	0x98110009	ERR_BADTASKID	Invalid task ID.
0xA	10	0x9811000A	ERR_NOIO	No IO.
0xB	11	0x9811000B	ERR_UNKNOWNAMSCMD	Unknown AMS command.
0xC	12	0x9811000C	ERR_WIN32ERROR	Win32 error.
0xD	13	0x9811000D	ERR_PORTNOTCONNECTED	Port not connected.
0xE	14	0x9811000E	ERR_INVALIDAMSLENGTH	Invalid AMS length.
0xF	15	0x9811000F	ERR_INVALIDAMSNETID	Invalid AMS Net ID.
0x10	16	0x98110010	ERR_LOWINSTLEVEL	Installation level is too low –TwinCAT 2 license error.
0x11	17	0x98110011	ERR_NODEBUGINTAVAILABLE	No debugging available.
0x12	18	0x98110012	ERR_PORTDISABLED	Port disabled – TwinCAT system service not started.
0x13	19	0x98110013	ERR_PORTALREADYCONNECTED	Port already connected.
0x14	20	0x98110014	ERR_AMSSYNC_W32ERROR	AMS Sync Win32 error.
0x15	21	0x98110015	ERR_AMSSYNC_TIMEOUT	AMS Sync Timeout.
0x16	22	0x98110016	ERR_AMSSYNC_AMSERROR	AMS Sync error.
0x17	23	0x98110017	ERR_AMSSYNC_NOINDEXINMAP	No index map for AMS Sync available.
0x18	24	0x98110018	ERR_INVALIDAMSPORT	Invalid AMS port.
0x19	25	0x98110019	ERR_NOMEMORY	No memory.
0x1A	26	0x9811001A	ERR_TCPSEND	TCP send error.
0x1B	27	0x9811001B	ERR_HOSTUNREACHABLE	Host unreachable.
0x1C	28	0x9811001C	ERR_INVALIDAMSFRAGMENT	Invalid AMS fragment.
0x1D	29	0x9811001D	ERR_TLSSEND	TLS send error – secure ADS connection failed.
0x1E	30	0x9811001E	ERR_ACCESSDENIED	Access denied – secure ADS access denied.

Router error codes

Hex	Dec	HRESULT	Name	Description
0x500	1280	0x98110500	ROUTERERR_NOLOCKEDMEMORY	Locked memory cannot be allocated.
0x501	1281	0x98110501	ROUTERERR_RESIZEMEMORY	The router memory size could not be changed.
0x502	1282	0x98110502	ROUTERERR_MAILBOXFULL	The mailbox has reached the maximum number of possible messages.
0x503	1283	0x98110503	ROUTERERR_DEBUGBOXFULL	The Debug mailbox has reached the maximum number of possible messages.
0x504	1284	0x98110504	ROUTERERR_UNKNOWNPORTTYPE	The port type is unknown.
0x505	1285	0x98110505	ROUTERERR_NOTINITIALIZED	The router is not initialized.
0x506	1286	0x98110506	ROUTERERR_PORTALREADYINUSE	The port number is already assigned.

Hex	Dec	HRESULT	Name	Description
0x507	1287	0x98110507	ROUTERERR_NOTREGISTERED	The port is not registered.
0x508	1288	0x98110508	ROUTERERR_NOMOREQUEUES	The maximum number of ports has been reached.
0x509	1289	0x98110509	ROUTERERR_INVALIDPORT	The port is invalid.
0x50A	1290	0x9811050A	ROUTERERR_NOTACTIVATED	The router is not active.
0x50B	1291	0x9811050B	ROUTERERR_FRAGMENTBOXFULL	The mailbox has reached the maximum number for fragmented messages.
0x50C	1292	0x9811050C	ROUTERERR_FRAGMENTTIMEOUT	A fragment timeout has occurred.
0x50D	1293	0x9811050D	ROUTERERR_TOBEREMOVED	The port is removed.

General ADS error codes

Hex	Dec	HRESULT	Name	Description
0x700	1792	0x98110700	ADSERR_DEVICE_ERROR	General device error.
0x701	1793	0x98110701	ADSERR_DEVICE_SRVNOTSUPP	Service is not supported by the server.
0x702	1794	0x98110702	ADSERR_DEVICE_INVALIDGRP	Invalid index group.
0x703	1795	0x98110703	ADSERR_DEVICE_INVALIDOFFSET	Invalid index offset.
0x704	1796	0x98110704	ADSERR_DEVICE_INVALIDACCESS	Reading or writing not permitted.
0x705	1797	0x98110705	ADSERR_DEVICE_INVALIDSIZE	Parameter size not correct.
0x706	1798	0x98110706	ADSERR_DEVICE_INVALIDDATA	Invalid data values.
0x707	1799	0x98110707	ADSERR_DEVICE_NOTREADY	Device is not ready to operate.
0x708	1800	0x98110708	ADSERR_DEVICE_BUSY	Device is busy.
0x709	1801	0x98110709	ADSERR_DEVICE_INVALIDCONTEXT	Invalid operating system context. This can result from use of ADS blocks in different tasks. It may be possible to resolve this through multitasking synchronization in the PLC.
0x70A	1802	0x9811070A	ADSERR_DEVICE_NOMEMORY	Insufficient memory.
0x70B	1803	0x9811070B	ADSERR_DEVICE_INVALIDPARM	Invalid parameter values.
0x70C	1804	0x9811070C	ADSERR_DEVICE_NOTFOUND	Not found (files, ...).
0x70D	1805	0x9811070D	ADSERR_DEVICE_SYNTAX	Syntax error in file or command.
0x70E	1806	0x9811070E	ADSERR_DEVICE_INCOMPATIBLE	Objects do not match.
0x70F	1807	0x9811070F	ADSERR_DEVICE_EXISTS	Object already exists.
0x710	1808	0x98110710	ADSERR_DEVICE_SYMBOLNOTFOUND	Symbol not found.
0x711	1809	0x98110711	ADSERR_DEVICE_SYMBOLVERSIONINVALID	Invalid symbol version. This can occur due to an online change. Create a new handle.
0x712	1810	0x98110712	ADSERR_DEVICE_INVALIDSTATE	Device (server) is in invalid state.
0x713	1811	0x98110713	ADSERR_DEVICE_TRANSMODENOTSUPP	AdsTransMode not supported.
0x714	1812	0x98110714	ADSERR_DEVICE_NOTIFYHNDINVALID	Notification handle is invalid.
0x715	1813	0x98110715	ADSERR_DEVICE_CLIENTUNKNOWN	Notification client not registered.
0x716	1814	0x98110716	ADSERR_DEVICE_NOMOREHDLS	No further handle available.
0x717	1815	0x98110717	ADSERR_DEVICE_INVALIDWATCHSIZE	Notification size too large.
0x718	1816	0x98110718	ADSERR_DEVICE_NOTINIT	Device not initialized.
0x719	1817	0x98110719	ADSERR_DEVICE_TIMEOUT	Device has a timeout.
0x71A	1818	0x9811071A	ADSERR_DEVICE_NOINTERFACE	Interface query failed.
0x71B	1819	0x9811071B	ADSERR_DEVICE_INVALIDINTERFACE	Wrong interface requested.
0x71C	1820	0x9811071C	ADSERR_DEVICE_INVALIDCLSID	Class ID is invalid.
0x71D	1821	0x9811071D	ADSERR_DEVICE_INVALIDOBJID	Object ID is invalid.
0x71E	1822	0x9811071E	ADSERR_DEVICE_PENDING	Request pending.
0x71F	1823	0x9811071F	ADSERR_DEVICE_ABORTED	Request is aborted.
0x720	1824	0x98110720	ADSERR_DEVICE_WARNING	Signal warning.
0x721	1825	0x98110721	ADSERR_DEVICE_INVALIDARRAYIDX	Invalid array index.
0x722	1826	0x98110722	ADSERR_DEVICE_SYMBOLNOTACTIVE	Symbol not active.
0x723	1827	0x98110723	ADSERR_DEVICE_ACCESSDENIED	Access denied.
0x724	1828	0x98110724	ADSERR_DEVICE_LICENSENOTFOUND	Missing license.
0x725	1829	0x98110725	ADSERR_DEVICE_LICENSEEXPIRED	License expired.
0x726	1830	0x98110726	ADSERR_DEVICE_LICENSEEXCEEDED	License exceeded.
0x727	1831	0x98110727	ADSERR_DEVICE_LICENSEINVALID	Invalid license.
0x728	1832	0x98110728	ADSERR_DEVICE_LICENSESYSTEMID	License problem: System ID is invalid.
0x729	1833	0x98110729	ADSERR_DEVICE_LICENSENOTIMELIMIT	License not limited in time.
0x72A	1834	0x9811072A	ADSERR_DEVICE_LICENSEFUTUREISSUE	Licensing problem: time in the future.
0x72B	1835	0x9811072B	ADSERR_DEVICE_LICENSETIMETOLONG	License period too long.

Hex	Dec	HRESULT	Name	Description
0x72C	1836	0x9811072C	ADSERR_DEVICE_EXCEPTION	Exception at system startup.
0x72D	1837	0x9811072D	ADSERR_DEVICE_LICENSEDUPLICATED	License file read twice.
0x72E	1838	0x9811072E	ADSERR_DEVICE_SIGNATUREINVALID	Invalid signature.
0x72F	1839	0x9811072F	ADSERR_DEVICE_CERTIFICATEINVALID	Invalid certificate.
0x730	1840	0x98110730	ADSERR_DEVICE_LICENSEOEMNOTFOUND	Public key not known from OEM.
0x731	1841	0x98110731	ADSERR_DEVICE_LICENSERESTRICTED	License not valid for this system ID.
0x732	1842	0x98110732	ADSERR_DEVICE_LICENSEDEMODENIED	Demo license prohibited.
0x733	1843	0x98110733	ADSERR_DEVICE_INVALIDFNCID	Invalid function ID.
0x734	1844	0x98110734	ADSERR_DEVICE_OUTOFRANGE	Outside the valid range.
0x735	1845	0x98110735	ADSERR_DEVICE_INVALIDALIGNMENT	Invalid alignment.
0x736	1846	0x98110736	ADSERR_DEVICE_LICENSEPLATFORM	Invalid platform level.
0x737	1847	0x98110737	ADSERR_DEVICE_FORWARD_PL	Context – forward to passive level.
0x738	1848	0x98110738	ADSERR_DEVICE_FORWARD_DL	Context – forward to dispatch level.
0x739	1849	0x98110739	ADSERR_DEVICE_FORWARD_RT	Context – forward to real time.
0x740	1856	0x98110740	ADSERR_CLIENT_ERROR	Client error.
0x741	1857	0x98110741	ADSERR_CLIENT_INVALIDPARM	Service contains an invalid parameter.
0x742	1858	0x98110742	ADSERR_CLIENT_LISTEMPTY	Polling list is empty.
0x743	1859	0x98110743	ADSERR_CLIENT_VARUSED	Var connection already in use.
0x744	1860	0x98110744	ADSERR_CLIENT_DUPLINVOKEID	The called ID is already in use.
0x745	1861	0x98110745	ADSERR_CLIENT_SYNCTIMEOUT	Timeout has occurred – the remote terminal is not responding in the specified ADS timeout. The route setting of the remote terminal may be configured incorrectly.
0x746	1862	0x98110746	ADSERR_CLIENT_W32ERROR	Error in Win32 subsystem.
0x747	1863	0x98110747	ADSERR_CLIENT_TIMEOUTINVALID	Invalid client timeout value.
0x748	1864	0x98110748	ADSERR_CLIENT_PORTNOTOPEN	Port not open.
0x749	1865	0x98110749	ADSERR_CLIENT_NOAMSADDR	No AMS address.
0x750	1872	0x98110750	ADSERR_CLIENT_SYNCINTERNAL	Internal error in Ads sync.
0x751	1873	0x98110751	ADSERR_CLIENT_ADDHASH	Hash table overflow.
0x752	1874	0x98110752	ADSERR_CLIENT_REMOVEHASH	Key not found in the table.
0x753	1875	0x98110753	ADSERR_CLIENT_NOMORESYM	No symbols in the cache.
0x754	1876	0x98110754	ADSERR_CLIENT_SYNCRESINVALID	Invalid response received.
0x755	1877	0x98110755	ADSERR_CLIENT_SYNCPORTLOCKED	Sync Port is locked.
0x756	1878	0x98110756	ADSERR_CLIENT_REQUESTCANCELLED	The request was cancelled.

RTime error codes

Hex	Dec	HRESULT	Name	Description
0x1000	4096	0x98111000	RTERR_INTERNAL	Internal error in the real-time system.
0x1001	4097	0x98111001	RTERR_BADTIMERPERIODS	Timer value is not valid.
0x1002	4098	0x98111002	RTERR_INVALIDTASKPTR	Task pointer has the invalid value 0 (zero).
0x1003	4099	0x98111003	RTERR_INVALIDSTACKPTR	Stack pointer has the invalid value 0 (zero).
0x1004	4100	0x98111004	RTERR_PRIOEXISTS	The request task priority is already assigned.
0x1005	4101	0x98111005	RTERR_NOMORETCB	No free TCB (Task Control Block) available. The maximum number of TCBs is 64.
0x1006	4102	0x98111006	RTERR_NOMORESEMAS	No free semaphores available. The maximum number of semaphores is 64.
0x1007	4103	0x98111007	RTERR_NOMOREQUEUES	No free space available in the queue. The maximum number of positions in the queue is 64.
0x100D	4109	0x9811100D	RTERR_EXTIRQALREADYDEF	An external synchronization interrupt is already applied.
0x100E	4110	0x9811100E	RTERR_EXTIRQNOTDEF	No external sync interrupt applied.
0x100F	4111	0x9811100F	RTERR_EXTIRQINSTALLFAILED	Application of the external synchronization interrupt has failed.
0x1010	4112	0x98111010	RTERR_IRQLNOTLESSOREQUAL	Call of a service function in the wrong context
0x1017	4119	0x98111017	RTERR_VMXNOTSUPPORTED	Intel VT-x extension is not supported.
0x1018	4120	0x98111018	RTERR_VMXDISABLED	Intel VT-x extension is not enabled in the BIOS.
0x1019	4121	0x98111019	RTERR_VMXCONTROLSMISSING	Missing function in Intel VT-x extension.
0x101A	4122	0x9811101A	RTERR_VMXENABLEFAILS	Activation of Intel VT-x fails.

Specific positive HRESULT Return Codes:

HRESULT	Name	Description
0x0000_0000	S_OK	No error.
0x0000_0001	S_FALSE	No error. Example: successful processing, but with a negative or incomplete result.
0x0000_0203	S_PENDING	No error. Example: successful processing, but no result is available yet.
0x0000_0256	S_WATCHDOG_TIMEOUT	No error. Example: successful processing, but a timeout occurred.

TCP Winsock error codes

Hex	Dec	Name	Description
0x274C	10060	WSAETIMEDOUT	A connection timeout has occurred - error while establishing the connection, because the remote terminal did not respond properly after a certain period of time, or the established connection could not be maintained because the connected host did not respond.
0x274D	10061	WSAECONNREFUSED	Connection refused - no connection could be established because the target computer has explicitly rejected it. This error usually results from an attempt to connect to a service that is inactive on the external host, that is, a service for which no server application is running.
0x2751	10065	WSAEHOSTUNREACH	No route to host - a socket operation referred to an unavailable host.
More Winsock error codes: Win32 error codes			

More Information:
www.beckhoff.com/automation

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com