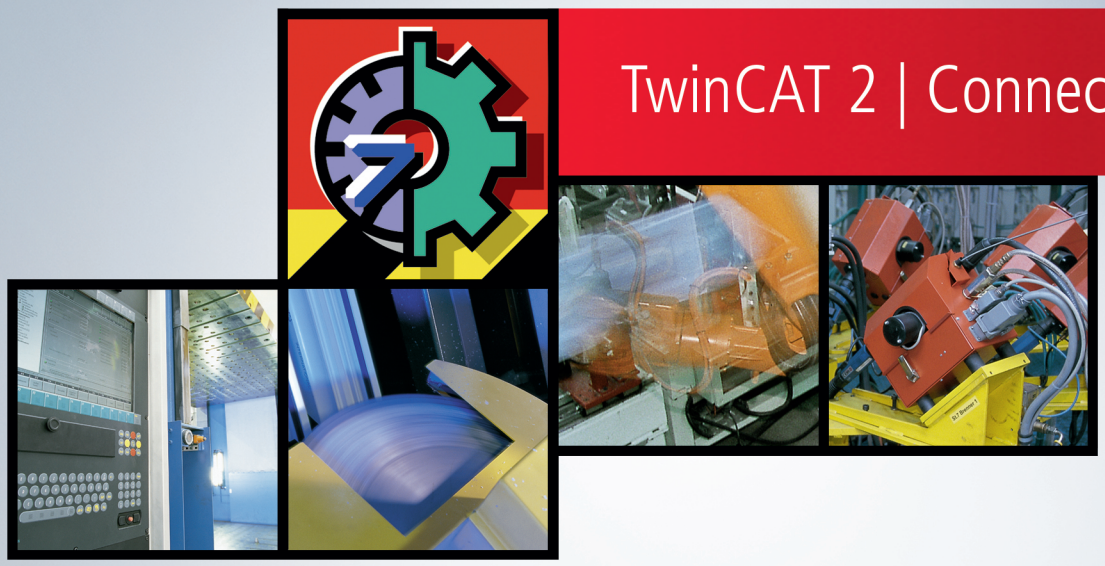
BECKHOFF New Automation Technology

Handbuch | DE

TX1000

TwinCAT 2 | ADS-Script-DLL



04.08.2022 | Version: 1.0

Inhaltsverzeichnis

1 Vorwort 5
1.1 Hinweise zur Dokumentation 5
1.2 Sicherheitshinweise 6
1.3 Hinweise zur Informationssicherheit 7
2 Einleitung 8
3 API 9
3.1 Methoden - allgemein 10
3.1.1 ConnectTo 10
3.1.2 ReadAdsState 10
3.1.3 WriteAdsState 11
3.2 Methoden - synchron 12
3.2.1 ReadVar 12
3.2.2 ReadBool 12
3.2.3 ReadInt8 13
3.2.4 ReadInt16 13
3.2.5 ReadInt32 14
3.2.6 ReadReal32 14
3.2.7 ReadReal64 15
3.2.8 ReadArrayOfBool 16
3.2.9 ReadArrayOfInt8 16
3.2.10 ReadArrayOfInt16 17
3.2.11 ReadArrayOfInt32 17
3.2.12 ReadArrayOfReal32 18
3.2.13 ReadArrayOfReal64 19
3.2.14 WriteVar 19
3.2.15 WriteBool 20
3.2.16 WriteInt8 20
3.2.17 WriteInt16 21
3.2.18 WriteInt32 21
3.2.19 WriteReal32 22
3.2.20 WriteReal64 23
3.2.21 WriteArrayOfBool 23
3.2.22 WriteArrayOfInt8 24
3.2.23 WriteArrayOfInt16 24
3.2.24 WriteArrayOfInt32 25
3.2.25 WriteArrayOfReal32 26
3.2.26 WriteArrayOfReal64 26
4 Beispiele 28
4.1 Windows Scripting Host 28
4.2 Visual Basic: Lesen und Schreiben von SPS Variablen 30
4.3 Active Server Pages 33
4.4 Active Server Pages für Windows CE 35
5 ADS Return Codes 38

1 Vorwort

1.1 Hinweise zur Dokumentation

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.
Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.
Das Fachpersonal ist verpflichtet, für jede Installation und Inbetriebnahme die zu dem betreffenden Zeitpunkt veröffentliche Dokumentation zu verwenden.

Das Fachpersonal hat sicherzustellen, dass die Anwendung bzw. der Einsatz der beschriebenen Produkte alle Sicherheitsanforderungen, einschließlich sämtlicher anwendbaren Gesetze, Vorschriften, Bestimmungen und Normen erfüllt.

Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiter entwickelt.
Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.
Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Marken

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH.
Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.

EtherCAT®

EtherCAT® ist eine eingetragene Marke und patentierte Technologie lizenziert durch die Beckhoff Automation GmbH, Deutschland

Copyright

1.2 Sicherheitshinweise

Sicherheitsbestimmungen

Beachten Sie die folgenden Sicherheitshinweise und Erklärungen!
Produktspezifische Sicherheitshinweise finden Sie auf den folgenden Seiten oder in den Bereichen Montage, Verdrahtung, Inbetriebnahme usw.

Haftungsausschluss

Die gesamten Komponenten werden je nach Anwendungsbestimmungen in bestimmten Hard- und Software-Konfigurationen ausgeliefert. Änderungen der Hard- oder Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig und bewirken den Haftungsausschluss der Beckhoff Automation GmbH & Co. KG.

Qualifikation des Personals

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs-, Automatisierungs- und Antriebstechnik, das mit den geltenden Normen vertraut ist.

Erklärung der Symbole

In der vorliegenden Dokumentation werden die folgenden Symbole mit einem nebenstehenden Sicherheitshinweis oder Hinweistext verwendet. Die Sicherheitshinweise sind aufmerksam zu lesen und unbedingt zu befolgen!

⚠ GEFAHR

Akute Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht unmittelbare Gefahr für Leben und Gesundheit von Personen!

⚠ WARNUNG

Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht Gefahr für Leben und Gesundheit von Personen!

⚠ VORSICHT

Schädigung von Personen!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, können Personen geschädigt werden!

HINWEIS

Schädigung von Umwelt oder Geräten

Wenn der Hinweis neben diesem Symbol nicht beachtet wird, können Umwelt oder Geräte geschädigt werden.

Tipp oder Fingerzeig

Dieses Symbol kennzeichnet Informationen, die zum besseren Verständnis beitragen.

1.3 Hinweise zur Informationssicherheit

Die Produkte der Beckhoff Automation GmbH & Co. KG (Beckhoff) sind, sofern sie online zu erreichen sind, mit Security-Funktionen ausgestattet, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen. Trotz der Security-Funktionen sind die Erstellung, Implementierung und ständige Aktualisierung eines ganzheitlichen Security-Konzepts für den Betrieb notwendig, um die jeweilige Anlage, das System, die Maschine und die Netzwerke gegen Cyber-Bedrohungen zu schützen. Die von Beckhoff verkauften Produkte bilden dabei nur einen Teil des gesamtheitlichen Security-Konzepts. Der Kunde ist dafür verantwortlich, dass unbefugte Zugriffe durch Dritte auf seine Anlagen, Systeme, Maschinen und Netzwerke verhindert werden. Letztere sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn entsprechende Schutzmaßnahmen eingerichtet wurden.

Zusätzlich sollten die Empfehlungen von Beckhoff zu entsprechenden Schutzmaßnahmen beachtet werden. Weiterführende Informationen über Informationssicherheit und Industrial Security finden Sie in unserem https://www.beckhoff.de/secguide.

Die Produkte und Lösungen von Beckhoff werden ständig weiterentwickelt. Dies betrifft auch die Security-Funktionen. Aufgrund der stetigen Weiterentwicklung empfiehlt Beckhoff ausdrücklich, die Produkte ständig auf dem aktuellen Stand zu halten und nach Bereitstellung von Updates diese auf die Produkte aufzuspielen. Die Verwendung veralteter oder nicht mehr unterstützter Produktversionen kann das Risiko von Cyber-Bedrohungen erhöhen.

Um stets über Hinweise zur Informationssicherheit zu Produkten von Beckhoff informiert zu sein, abonnieren Sie den RSS Feed unter https://www.beckhoff.de/secinfo.

2 Einleitung

Mit der ADS-Script-DLL kann von Scriptsprachen (VBScript oder JScript) auf ADS-Geräte zugegriffen werden. Verwendung finden die Scriptsprachen beim Windows Scripting Host (WSH) oder bei der Erstellung von interaktiven Internetanwendungen. Die Scripte können sowohl auf dem Webserver (Active Server Pages), als auch auf dem Client (DHTML, ...) ausgeführt werden.

Da die Scriptsprachen nur wenige Datentypen unterstützen, ist für jeden möglichen Datentyp im ADS-Gerät eine entsprechende Methode vorhanden. Die Anzahl der Parameter wurde dabei auf ein Minimum reduziert. Eine Ausnahme bilden die Methoden, bei denen per Name auf die Variable zugegriffen wird (TcScriptSync::ReadVar() [▶ 12] und TcScriptSync::WriteVar() [▶ 19]). Hierbei wird der Datentyp der ADS-Variable aus dem ADS-Gerät ermittelt. Die beiden Methoden wandeln intern die Parameter in die entsprechenden Datentypen um und greifen anschließend auf das ADS-Gerät zu.

3 API

Da die Scriptsprachen nur wenige Datentypen unterstützen, ist für jeden möglichen Datentyp im ADS-Gerät eine entsprechende Methode vorhanden. Die Anzahl der Parameter wurde dabei auf ein Minimum reduziert. Eine Ausnahme bilden die Methoden, bei denen per Name auf die Variable zugegriffen wird (TcScriptSync::ReadVar() [▶ 12] und TcScriptSync::WriteVar() [▶ 19]). Hierbei wird der Datentyp der ADS-Variable aus dem ADS-Gerät ermittelt. Die beiden Methoden wandeln intern die Parameter in die entsprechenden Datentypen um und greifen anschließend auf das ADS-Gerät zu.

Verwendung		**Methode**	
allgemein	**Kommunikation**	TcScriptSync::ConnectTo() [▶ 10]	
	Lesen	TcScriptSync::ReadAdsState() [▶ 10]	
	Schreiben	TcScriptSync::WriteAdsState() [▶ 11]	
		per Adresse	**per Variablenname**
synchron	**Lesen**	TcScriptSync::ReadBool() [▶ 12] TcScriptSync::ReadInt8() [▶ 13] TcScriptSync::ReadInt16() [▶ 13] TcScriptSync::ReadInt32() [▶ 14] TcScriptSync::ReadReal32() [▶ 14] TcScriptSync::ReadReal64() [▶ 15] TcScriptSync::ReadArrayOfBool() [▶ 16] TcScriptSync::ReadArrayOfInt8() [▶ 16] TcScriptSync::ReadArrayOfInt16() [▶ 17] TcScriptSync::ReadArrayOfInt32() [▶ 17] TcScriptSync::ReadArrayOfReal32() [▶ 18] TcScriptSync::ReadArrayOfReal64() [▶ 19]	TcScriptSync::ReadVar() [▶ 12]
	Schreiben	TcScriptSync::WriteBool() [▶ 20] TcScriptSync::WriteInt8() [▶ 20] TcScriptSync::WriteInt16() [▶ 21] TcScriptSync::WriteInt32() [▶ 21] TcScriptSync::WriteReal32() [▶ 22] TcScriptSync::WriteReal64() [▶ 23] TcScriptSync::WriteArrayOfBool() [▶ 23] TcScriptSync::WriteArrayOfInt8() [▶ 24] TcScriptSync::WriteArrayOfInt16() [▶ 24] TcScriptSync::WriteArrayOfInt32() [▶ 25] TcScriptSync::WriteArrayOfReal32() [▶ 26] TcScriptSync::WriteArrayOfReal64() [▶ 26]	TcScriptSync::WriteVar() [▶ 19]

3.1 Methoden - allgemein

3.1.1 ConnectTo

Öffnet einen Kommunikationskanal zwischen dem Objekt und einem ADS-Gerät (SPS, NC, ...).

```
object.ConnectTo(
  sAmsNetId As String,
  nPort As Long
)
```

Parameter

sAmsNetId	AdsAmsNetId.
nPort	Portnummer.

Rückgabewert

-

Beschreibung

Der erste Parameter enthält die sechsstellige AdsAmsNetId. Die einzelnen Werte werden durch einen Punkt voneinander getrennt. Wird ein leerer String übergeben, so wird die lokale AdsAmsNetId benutzt.

Bestand vor dem Aufruf von *ConnectTo()* schon ein geöffneter Kommunikationskanal, so wird dieser automatisch geschlossen. Der Kommunikationskanal wird ebenfalls geschlossen, wenn das Objekt gelöscht wird.

Beispiele

VBScript:

```
Dim TcClientSync
Set TcClientSync =
CreateObject("TCSCRIPT.TcScriptSync")
Call TcClientSync.ConnectTo("", 801)
```

JScript:

```
var TcClientSync;
TcClientSync = new
ActiveXObject("TCSCRIPT.TcScriptSync");
TcClientSync.ConnectTo("172.16.17.13.1.1", 811);
```

3.1.2 ReadAdsState

Liest den aktuellen ADS-Zustand von einem ADS-Gerät aus.

```
object.ReadAdsState() As Long
```

Parameter

-

Rückgabewert

ADS-Status.

Beschreibung

Jedes ADS-Gerät kann verschiedene Zustände annehmen. So kann z.B. die SPS im STOP- oder im RUN-Zustand sein. Dieser Zustand lässt sich mit Hilfe der Methode *ReadAdsState()* abfragen. Eine Übersicht aller möglichen Zustände finden Sie z.B. bei der Beschreibung des Enums *ADSSTATE* vom ADS-OCX. Bei der Auswertung des ADS-Zustandes ist zu beachten, dass nicht jedes ADS-Gerät jeden Zustand annehmen kann. So kann sich z.B. die NC nicht im STOP-Zustand befinden.
Die wichtigsten ADS-Zustände sind RUN (5) und STOP (6) und werden am häufigsten im Zusammenhang mit der SPS eingesetzt.

Beispiele

VBScript:

```
Dim nAdsState

nAdsState = TcClientSync.ReadAdsState()

WScript.echo "AdsState = ", nAdsState
```

JScript:

```
var nAdsState;

nAdsState = TcClientSync.ReadAdsState();

WScript.echo("AdsState = ", nAdsState);
```

3.1.3 WriteAdsState

Ändert den ADS-Status eines ADS-Gerätes.

```
object.WriteAdsState(nAdsState As Long)
```

Parameter

nAdsState	neuer ADS-Status.

Rückgabewert

-

Beschreibung

Jedes ADS-Gerät kann verschiedene Zustände annehmen. So kann z.B. die SPS im STOP- oder im RUN-Zustand sein. Dieser Zustand lässt sich mit Hilfe der Methode *WriteAdsState()* verändern. Eine Übersicht aller möglichen Zustände finden Sie z.B. bei der Beschreibung des Enums *ADSSTATE* vom ADS-OCX. Bei der Auswertung des ADS-Zustandes ist zu beachten, dass nicht jedes ADS-Gerät jeden Zustand annehmen kann. So kann sich z.B. die NC nicht im STOP-Zustand befinden.
Die wichtigsten ADS-Zustände sind RUN (5) und STOP (6) und werden am häufigsten im Zusammenhang mit der SPS eingesetzt.

Beispiele

VBScript:

```
Call TcClientSync.WriteAdsState(5)
```

JScript:

```
TcClientSync.WriteAdsState(5);
```

3.2 Methoden - synchron

3.2.1 ReadVar

Liest den Wert einer Variable aus einem ADS-Gerät.

```
object.ReadVar(
  sVarName As String
) As Variant
```

Parameter

sVarName	Name der ADS-Variable.

Rückgabewert

Wert der ADS-Variable.

Beschreibung

Der Lesevorgang wird synchron durchgeführt.

Beispiele

VBScript:

```
Dim VarValue
VarValue = TcClientSync.ReadVar(".PLCVarSInt")
WScript.echo "VarValue = ", VarValue
```

JScript:

```
var VarValue;
VarValue = TcClientSync.ReadVar(".PLCVarSInt");
WScript.echo("VarValue = ", VarValue);
```

3.2.2 ReadBool

Liest eine Variable vom Typ Boolean aus einem ADS-Gerät.

```
object.ReadBool(
  nIndexGroup As Long,
  nIndexOffset As Long
) As Boolean
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.

Rückgabewert

Wert der ADS-Variable.

Beschreibung

Der Lesevorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp BOOL.

Beispiele

VBScript:

```
Dim VarBool
VarBool = TcClientSync.ReadBool(&H4020, 0)
WScript.echo "VarBool = ", VarBool
```

JScript:

```
var VarBool;
VarBool = TcClientSync.ReadBool(0x4020, 0);
WScript.echo("VarBool = ", VarBool);
```

3.2.3 ReadInt8

Liest eine Variable vom Typ Byte aus einem ADS-Gerät.

```
object.ReadInt8(
  nIndexGroup As Long,
  nIndexOffset As Long
) As Byte
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.

Rückgabewert

Wert der ADS-Variable.

Beschreibung

Der Lesevorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp BYTE, SINT oder USINT.

Beispiele

VBScript:

```
Dim VarInt8
VarInt8 = TcClientSync.ReadInt8(&H4020, 0)
WScript.echo "VarInt8 = ", VarInt8
```

JScript:

```
var VarInt8;
VarInt8 = TcClientSync.ReadInt8(0x4020, 0);
WScript.echo("VarInt8 = ", VarInt8);
```

3.2.4 ReadInt16

Liest eine Variable vom Typ Integer aus einem ADS-Gerät.

```
object.ReadInt16(
  nIndexGroup  As Long,
  nIndexOffset As Long
) As Integer
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.

Rückgabewert

Wert der ADS-Variable.

Beschreibung

Der Lesevorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp WORD, INT oder UINT.

Beispiele

VBScript:

```
Dim VarInt16
VarInt16 = TcClientSync.ReadInt16(&H4020, 0)
WScript.echo "VarInt16 = ", VarInt16
```

JScript:

```
var VarInt16;
VarInt16 = TcClientSync.ReadInt16(0x4020, 0);
WScript.echo("VarInt16 = ", VarInt16);
```

3.2.5 ReadInt32

Liest eine Variable vom Typ Long aus einem ADS-Gerät.

```
object.ReadInt32(
  nIndexGroup As Long,
  nIndexOffset As Long
) As Long
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.

Rückgabewert

Wert der ADS-Variable.

Beschreibung

Der Lesevorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp DWORD, DINT oder UDINT.

Beispiele

VBScript:

```
Dim VarInt32
VarInt32 = TcClientSync.ReadInt32(&H4020, 0)
WScript.echo "VarInt32 = ", VarInt32
```

JScript:

```
var VarInt32;
VarInt32 = TcClientSync.ReadInt32(0x4020, 0);
WScript.echo("VarInt32 = ", VarInt32);
```

3.2.6 ReadReal32

Liest eine Variable vom Typ Float aus einem ADS-Gerät.

```
object.ReadReal32(
  nIndexGroup As Long,
  nIndexOffset As Long
) As Single
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.

Rückgabewert

Wert der ADS-Variable.

Beschreibung

Der Lesevorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp REAL.

Beispiele

VBScript:

```
Dim VarReal32
VarReal32 = TcClientSync.ReadReal32(&H4020, 0)
WScript.echo "VarReal32 = ", VarReal32
```

JScript:

```
var VarReal32;
VarReal32 = TcClientSync.ReadReal32(0x4020, 0);
WScript.echo("VarReal32 = ", VarReal32);
```

3.2.7 ReadReal64

Liest eine Variable vom Typ Double aus einem ADS-Gerät.

```
object.ReadReal64(
  nIndexGroup As Long,
  nIndexOffset As Long
) As Double
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.

Rückgabewert

Wert der ADS-Variable.

Beschreibung

Der Lesevorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp LREAL.

Beispiele

VBScript:

```
Dim VarReal64
VarReal64 = TcClientSync.ReadReal64(&H4020, 0)
WScript.echo "VarReal64 = ", VarReal64
```

JScript:

```
var VarReal64;
VarReal64 = TcClientSync.ReadReal64(0x4020, 0);
WScript.echo("VarReal64 = ", VarReal64);
```

3.2.8 ReadArrayOfBool

Liest ein Array mit Elementen vom Typ Boolean aus einem ADS-Gerät.

```
object.ReadArrayOfBool(
  nIndexGroup As Long,
  nIndexOffset As Long,
  nCount As Long
) As Variant
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
nCount	Anzahl der Elemente, die gelesen werden sollen.

Rückgabewert

Array, mit den Werten aus dem ADS-Gerät.

Beschreibung

Der Lesevorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp ARRAY OF BOOL.

Beispiele

VBScript:

```
Dim VarArrayBool
VarArrayBool = TcClientSync.ReadArrayOfBool(&H4020, 0, 2)
WScript.echo "VarArrayBool(0) = ", VarArrayBool(0)
WScript.echo "VarArrayBool(1) = ", VarArrayBool(1)
```

JScript:

```
var VarArrayBool;
VarArrayBool = TcClientSync.ReadArrayOfBool(0x4020, 0, 2);
WScript.echo("VarArrayBool[0] = ",VarArrayBool[0]);
WScript.echo("VarArrayBool[1] = ", VarArrayBool[1]);
```

3.2.9 ReadArrayOfInt8

Liest ein Array mit Elementen vom Typ Byte aus einem ADS-Gerät.

```
object.ReadArrayOfInt8(
  nIndexGroup As Long,
  nIndexOffset As Long,
  nCount As Long
) As Variant
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
nCount	Anzahl der Elemente, die gelesen werden sollen.

Rückgabewert

Array, mit den Werten aus dem ADS-Gerät.

Beschreibung

Der Lesevorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp ARRAY OF BYTE, ARRAY OF SINT oder ARRAY OF USINT.

Beispiele

VBScript:

```
Dim VarArrayInt8
VarArrayInt8 = TcClientSync.ReadArrayOfInt8(&H4020, 0, 2)
WScript.echo "VarArrayInt8(0) = ", VarArrayInt8(0)
WScript.echo "VarArrayInt8(1) = ", VarArrayInt8(1)
```

JScript:

```
var VarArrayInt8;
VarArrayInt8 = TcClientSync.ReadArrayOfInt8(0x4020, 0, 2);
WScript.echo("VarArrayInt8[0] = ", VarArrayInt8[0]);
WScript.echo("VarArrayInt8[1] = ", VarArrayInt8[1]);
```

3.2.10 ReadArrayOfInt16

Liest ein Array mit Elementen vom Typ Integer aus einem ADS-Gerät.

```
object.ReadArrayOfInt16(
  nIndexGroup As Long,
  nIndexOffset As Long,
  nCount As Long
) As Variant
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
nCount	Anzahl der Elemente, die gelesen werden sollen.

Rückgabewert

Array, mit den Werten aus dem ADS-Gerät.

Beschreibung

Der Lesevorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp ARRAY OF WORD, ARRAY OF INT oder ARRAY OF UINT.

Beispiele

VBScript:

```
Dim VarArrayInt16
VarArrayInt16 = TcClientSync.ReadArrayOfInt16(&H4020, 0,2)
WScript.echo "VarArrayInt16(0) = ",VarArrayInt16(0)
WScript.echo "VarArrayInt16(1) = ", VarArrayInt16(1)
```

JScript:

```
var VarArrayInt16;
VarArrayInt16 = TcClientSync.ReadArrayOfInt16(0x4020, 0, 2);
WScript.echo("VarArrayInt16[0] = ",VarArrayInt16[0]);
WScript.echo("VarArrayInt16[1] = ",VarArrayInt16[1]);
```

3.2.11 ReadArrayOfInt32

Liest ein Array mit Elementen vom Typ Long aus einem ADS-Gerät.

```
object.ReadArrayOfInt32(
  nIndexGroup As Long,
  nIndexOffset As Long,
  nCount As Long
) As Variant
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
nCount	Anzahl der Elemente, die gelesen werden sollen.

Rückgabewert

Array, mit den Werten aus dem ADS-Gerät.

Beschreibung

Der Lesevorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp ARRAY OF DWORD, ARRAY OF DINT oder ARRAY OF UDINT.

Beispiele

VBScript:

```
Dim VarArrayInt32
VarArrayInt32 = TcClientSync.ReadArrayOfInt32(&H4020, 0,2)
WScript.echo "VarArrayInt32(0) = ",VarArrayInt32(0)
WScript.echo "VarArrayInt32(1) = ", VarArrayInt32(1)
```

JScript:

```
var VarArrayInt32;
VarArrayInt32 = TcClientSync.ReadArrayOfInt32(0x4020, 0, 2);
WScript.echo("VarArrayInt32[0] = ",VarArrayInt32[0]);
WScript.echo("VarArrayInt32[1] = ",VarArrayInt32[1]);
```

3.2.12 ReadArrayOfReal32

Liest ein Array mit Elementen vom Typ Float aus einem ADS-Gerät.

```
object.ReadArrayOfReal32(
  nIndexGroup As Long,
  nIndexOffset As Long,
  nCount As Long
) As Variant
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
nCount	Anzahl der Elemente, die gelesen werden sollen.

Rückgabewert

Array, mit den Werten aus dem ADS-Gerät.

Beschreibung

Der Lesevorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp ARRAY OF REAL.

Beispiele

VBScript:

```
Dim VarArrayReal32
VarArrayReal32 = TcClientSync.ReadArrayOfReal32(&H4020, 0,2)
WScript.echo "VarArrayReal32(0) = ",VarArrayReal32(0)
WScript.echo "VarArrayReal32(1) = ",VarArrayReal32(1)
```

JScript:

```
var VarArrayReal32;
VarArrayReal32 = TcClientSync.ReadArrayOfReal32(0x4020, 0,2);
WScript.echo("VarArrayReal32[0] = ",VarArrayReal32[0]);
WScript.echo("VarArrayReal32[1] = ",VarArrayReal32[1]);
```

3.2.13 ReadArrayOfReal64

Liest ein Array mit Elementen vom Typ Double aus einem ADS-Gerät.

```
object.ReadArrayOfReal64(
  nIndexGroup As Long,
  nIndexOffset As Long,
  nCount As Long
) As Variant
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
nCount	Anzahl der Elemente, die gelesen werden sollen.

Rückgabewert

Array, mit den Werten aus dem ADS-Gerät.

Beschreibung

Der Lesevorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp ARRAY OF LREAL.

Beispiele

VBScript:

```
Dim VarArrayReal64
VarArrayReal64 = TcClientSync.ReadArrayOfReal64(&H4020, 0,2)
WScript.echo "VarArrayReal64(0) = ",VarArrayReal64(0)
WScript.echo "VarArrayReal64(1) = ",VarArrayReal64(1)
```

JScript:

```
var VarArrayReal64;
VarArrayReal64 = TcClientSync.ReadArrayOfReal64(0x4020, 0,2);
WScript.echo("VarArrayReal64[0] = ",VarArrayReal64[0]);
WScript.echo("VarArrayReal64[1] = ",VarArrayReal64[1]);
```

3.2.14 WriteVar

Schreibt in eine Variable eines ADS-Gerätes.

```
object.WriteVar(
  sVarName As String,
  nValue As Variant
)
```

Parameter

sVarName	Name der ADS-Variable.
nValue	Wert, der in die Variable geschrieben werden soll.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt.

Beispiele

VBScript:

```
Dim VarValue = 0
Call TcClientSync.WriteVar(".PLCSIntVar", VarValue)
```

JScript:

```
var VarValue =0;
TcClientSync.WriteVar(".PLCSIntVar",VarValue);
```

3.2.15 WriteBool

Schreibt in eine Variable vom Typ Boolean.

```
object.WriteBool(
  nIndexGroup As Long,
  nIndexOffset As Long,
  bValue As Boolean
)
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
bValue	Wert, der in die Variable geschrieben werden soll.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp BOOL.

Beispiele

VBScript:

```
Dim VarBool = 0
Call TcClientSync.WriteBool(&H4020, 0, VarBool)
```

JScript:

```
var VarBool =0;
TcClientSync.WriteBool(0x4020, 0, VarBool);
```

3.2.16 WriteInt8

Schreibt in eine Variable vom Typ Byte.

```
object.WriteInt8(
  nIndexGroup As Long,
  nIndexOffset As Long,
  nValue As Byte
)
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.

nValue	Wert, der in die Variable geschrieben werden soll.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp BYTE, SINT oder USINT.

Beispiele

VBScript:

```
Dim VarInt8 = 0
Call TcClientSync.WriteInt8(&H4020, 0, VarInt8)
```

JScript:

```
var VarInt8 =0;
TcClientSync.WriteInt8(0x4020, 0, VarInt8);
```

3.2.17 WriteInt16

Schreibt in eine Variable vom Typ Integer.

```
object.WriteInt16(
  nIndexGroup As Long,
  nIndexOffset As Long,
  nValue As Integer
)
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
nValue	Wert, der in die Variable geschrieben werden soll.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp WORD, INT oder UINT.

Beispiele

VBScript:

```
Dim VarInt16 = 0
Call TcClientSync.WriteInt16(&H4020, 0, VarInt16)
```

JScript:

```
var VarInt16 =0;
TcClientSync.WriteInt16(0x4020, 0, VarInt16);
```

3.2.18 WriteInt32

Schreibt in eine Variable vom Typ Long.

```
object.WriteInt32(
  nIndexGroup As Long,
  nIndexOffset As Long,
  nValue As Long
)
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
nValue	Wert, der in die Variable geschrieben werden soll.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp DWORD, DINT oder UDINT.

Beispiele

VBScript:

```
Dim VarInt32 = 0
Call TcClientSync.WriteInt32(&H4020, 0, VarInt32)
```

JScript:

```
var VarInt32 =0;
TcClientSync.WriteInt32(0x4020, 0, VarInt32);
```

3.2.19 WriteReal32

Schreibt in eine Variable vom Typ Single.

```
object.WriteReal32(
  nIndexGroup As Long,
  nIndexOffset As Long,
  fValue As Single
)
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
fValue	Wert, der in die Variable geschrieben werden soll.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp REAL.

Beispiele

VBScript:

```
Dim VarReal32 = 0
Call TcClientSync.WriteReal32(&H4020, 0, VarReal32)
```

JScript:

```
var VarReal32 =0;
TcClientSync.WriteReal32(0x4020, 0, VarReal32);
```

3.2.20 WriteReal64

Schreibt in eine Variable vom Typ Double.

```
object.WriteReal64(
  nIndexGroup As Long,
  nIndexOffset As Long,
  fValue As Double
)
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
fValue	Wert, der in die Variable geschrieben werden soll.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp LREAL.

Beispiele

VBScript:

```
Dim VarReal64 = 0
Call TcClientSync.WriteReal64(&H4020, 0, VarReal64)
```

JScript:

```
var VarReal64 =0;
TcClientSync.WriteReal64(0x4020, 0, VarReal64);
```

3.2.21 WriteArrayOfBool

Schreibt ein Array mit Elementen vom Typ Boolean in ein ADS-Gerät.

```
object.WriteArrayOfBool(
  nIndexGroup As Long,
  nIndexOffset As Long,
  pValue As Variant
)
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
pValue	Array, mit Elementen vom Typ Boolean.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp ARRAY OF BOOL.

Beispiele

VBScript:

```
Dim VarArrayBool(1)
VarArrayBool(0) = True
VarArrayBool(1) = False
Call TcClientSync.WriteArrayOfBool(&H4020, 0, VarArrayBool)
```

JScript:

```
var
VarArrayBool[2];
VarArrayBool[0] =true;
VarArrayBool[1] =false;
TcClientSync.WriteArrayOfBool(0x4020, 0,VarArrayBool);
```

3.2.22 WriteArrayOfInt8

Schreibt ein Array mit Elementen vom Typ Byte in ein ADS-Gerät.

```
object.WriteArrayOfInt8(
  nIndexGroup As Long,
  nIndexOffset As Long,
  pValue As Variant
)
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
pValue	Array, mit Elementen vom Typ Byte.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp ARRAY OF BYTE, ARRAY OF SINT oder ARRAY OF USINT.

Beispiele

VBScript:

```
Dim VarArrayInt8(1)
VarArrayInt8(0) = 0
VarArrayInt8(1) = 1
Call TcClientSync.WriteArrayOfInt8(&H4020, 0, VarArrayInt8)
```

JScript:

```
var
VarArrayInt8[2];
VarArrayInt8[0] =0;
VarArrayInt8[1] =1;
TcClientSync.WriteArrayOfInt8(0x4020, 0,VarArrayInt8);
```

3.2.23 WriteArrayOfInt16

Schreibt ein Array mit Elementen vom Typ Integer in ein ADS-Gerät.

```
object.WriteArrayOfInt16(
  nIndexGroup As Long,
  nIndexOffset As Long,
  pValue As Variant
)
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
pValue	Array, mit Elementen vom Typ Integer.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp ARRAY OF WORD, ARRAY OF INT oder ARRAY OF UINT.

Beispiele

VBScript:

```
Dim VarArrayInt16(1)
VarArrayInt16(0) = 0
VarArrayInt16(1) = 1
Call TcClientSync.WriteArrayOfInt16(&H4020, 0,VarArrayInt16)
```

JScript:

```
var
VarArrayInt16[2];
VarArrayInt16[0] =0;
VarArrayInt16[1] =1;
TcClientSync.WriteArrayOfInt16(0x4020, 0,VarArrayInt16);
```

3.2.24 WriteArrayOfInt32

Schreibt ein Array mit Elementen vom Typ Long in ein ADS-Gerät.

```
object.WriteArrayOfInt32(
  nIndexGroup As Long,
  nIndexOffset As Long,
  pValue As Variant
)
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
pValue	Array, mit Elementen vom Typ Long.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp ARRAY OF DWORD, ARRAY OF DINT oder ARRAY OF UDINT.

Beispiele

VBScript:

```
Dim VarArrayInt32(1)
VarArrayInt32(0) = 0
VarArrayInt32(1) = 1
Call TcClientSync.WriteArrayOfInt32(&H4020, 0,VarArrayInt32)
```

JScript:

```
Var VarArrayInt32[2];
VarArrayInt32[0] = 0;
VarArrayInt32[1] = 1;
TcClientSync.WriteArrayOfInt32(0x4020, 0, VarArrayInt32);
```

3.2.25 WriteArrayOfReal32

Schreibt ein Array mit Elementen vom Typ Single in ein ADS-Gerät.

```
object.WriteArrayOfReal32(
  nIndexGroup As Long,
  nIndexOffset As Long,
  pValue As Variant
)
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
pValue	Array, mit Elementen vom Typ Single.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp ARRAY OF REAL.

Beispiele

VBScript:

```
Dim VarArrayReal32(1)
VarArrayReal32(0) = 0.1
VarArrayReal32(1) = 1.2
Call TcClientSync.WriteArrayOfReal32(&H4020, 0,VarArrayReal32)
```

JScript:

```
var
VarArrayReal32[2];
VarArrayReal32[0] =0.1;
VarArrayReal32[1] =1.2;
TcClientSync.WriteArrayOfReal32(0x4020, 0,VarArrayReal32);
```

3.2.26 WriteArrayOfReal64

Schreibt ein Array mit Elementen vom Typ Double in ein ADS-Gerät.

```
object.WriteArrayOfReal64(
  nIndexGroup As Long,
  nIndexOffset As Long,
  pValue As Variant
)
```

Parameter

nIndexGroup	Index-Gruppe der ADS-Variable.
nIndexOffset	Index-Offset der ADS-Variable.
pValue	Array, mit Elementen vom Typ Double.

Rückgabewert

-

Beschreibung

Der Schreibvorgang wird synchron durchgeführt. In der IEC 61131-6 entspricht dieses dem Datentyp ARRAY OF LREAL.

Beispiele

VBScript:

```
Dim VarArrayReal64(1)
VarArrayReal64(0) = 0.1
VarArrayReal64(1) = 1.2
Call TcClientSync.WriteArrayOfReal64(&H4020, 0,VarArrayReal64)
```

JScript:

```
var
VarArrayReal64[2];
VarArrayReal64[0] =0.1;
VarArrayReal64[1] =1.2;
TcClientSync.WriteArrayOfReal64(0x4020, 0,VarArrayReal64);
```

4 Beispiele

Beschreibung	Quelltexte
Beispiel 1: Windows Scripting Host [▶ 28]	https://infosys.beckhoff.com/content/1031/ tcscriptdll/ Resources/12459716107.zip
Beispiel 2: Visual Basic [▶ 30]	https://infosys.beckhoff.com/content/1031/ tcscriptdll/ Resources/12459720971.zip
Beispiel 3: Active Server Pages [▶ 33]	https://infosys.beckhoff.com/content/1031/ tcscriptdll/ Resources/12459711755.zip

4.1 Windows Scripting Host

Laden Sie das SPS-Programm in die lokale SPS auf Ihrem PC und starten Sie das SPS-Programm. Rufen Sie anschließend das Script auf.

Nachdem der Kommunikationskanal geöffnet wurde, wird überprüft, ob sich die SPS im RUN-Zustand befindet. Falls nicht, wird die SPS gestartet. Als nächstes werden alle SPS-Variablen ausgelesen und einzeln angezeigt. Anschließend werden alle Variablen erhöht und zur Kontrolle wieder ausgelesen und angezeigt. Zum Schluß wird die SPS wieder gestoppt.

VBScript:

```
Dim TcClientSync
Dim VarBool, VarInt8, VarInt16, VarInt32, VarReal32, VarReal64

'create TcScript-Object
Set TcClientSync = CreateObject("TCSCRIPT.TcScriptSync")
Call TcClientSync.ConnectTo("", 801)

if (TcClientSync.ReadAdsState() <> 5) then
  WScript.echo("PLC is not running!" & vbCrLf & "Start PLC.")
  Call TcClientSync.WriteAdsState(5)
end if

'read from PLC
VarBool = TcClientSync.ReadBool(&H4020, 0)
WScript.echo "VarBool = ", VarBool

VarInt8 = TcClientSync.ReadInt8(&H4020, 2)
WScript.echo "VarInt8 = ", VarInt8

VarInt16 = TcClientSync.ReadInt16(&H4020, 4)
WScript.echo "VarInt16 = ", VarInt16

VarInt32 = TcClientSync.ReadInt32(&H4020, 6)
WScript.echo "VarInt32 = ", VarInt32

VarReal32 = TcClientSync.ReadReal32(&H4020, 10)
WScript.echo "VarReal32 = ", VarReal32

VarReal64 = TcClientSync.ReadReal64(&H4020, 14)
WScript.echo "VarReal64 = ", VarReal64

'write to PLC
Call TcClientSync.WriteBool(&H4020, 0, NOT VarBool)
Call TcClientSync.WriteInt8(&H4020, 2, VarInt8 + 1)
Call TcClientSync.WriteInt16(&H4020, 4, VarInt16 +1)
Call TcClientSync.WriteInt32(&H4020, 6, VarInt32 + 1)
Call TcClientSync.WriteReal32(&H4020, 10, VarReal32 +1.1)
Call TcClientSync.WriteReal64(&H4020, 14, VarReal64 +1.11)

Call TcClientSync.WriteVar(".PLCBoolVar", VarBool)
Call TcClientSync.WriteVar(".PLCSIntVar", VarInt8 +2)
Call TcClientSync.WriteVar(".PLCIntVar", VarInt16 +2)
Call TcClientSync.WriteVar(".PLCDIntVar", VarInt32 +2)
Call TcClientSync.WriteVar(".PLCRealVar", VarReal32 +2.1)
Call TcClientSync.WriteVar(".PLCLRealVar", VarReal64 +2.11)

'read again from PLC
```

```
VarBool = TcClientSync.ReadVar(".PLCBoolVar")
WScript.echo "VarBool = ", VarBool

VarInt8 = TcClientSync.ReadVar(".PLCSIntVar")
WScript.echo "VarInt8 = ", VarInt8

VarInt16 = TcClientSync.ReadVar(".PLCIntVar")
WScript.echo "VarInt16 = ", VarInt16

VarInt32 = TcClientSync.ReadVar(".PLCDIntVar")
WScript.echo "VarInt32 = ", VarInt32

VarReal32 = TcClientSync.ReadVar(".PLCRealVar")
WScript.echo "VarReal32 = ", VarReal32

VarReal64 = TcClientSync.ReadVar(".PLCLRealVar")
WScript.echo "VarReal64 = ", VarReal64

if (TcClientSync.ReadAdsState() = 5) then
  WScript.echo("PLC is running!" & vbCrLf & "Stop PLC.")
  Call TcClientSync.WriteAdsState(6)
end if
```

JScript:

```
var TcClientSync;
var VarBool, VarInt8, VarInt16, VarInt32, VarReal32, VarReal64;

// create TcScript-Object
TcClientSync = new ActiveXObject("TCSCRIPT.TcScriptSync");
TcClientSync.ConnectTo("", 801);

if (TcClientSync.ReadAdsState() != 5)
{
  WScript.echo("PLC is not running!\nStart PLC.");
  TcClientSync.WriteAdsState(5);
}
// read from PLC
VarBool = TcClientSync.ReadBool(0x4020, 0);
WScript.echo("VarBool = ", VarBool);

VarInt8 = TcClientSync.ReadInt8(0x4020, 2);
WScript.echo("VarInt8 = ", VarInt8);

VarInt16 = TcClientSync.ReadInt16(0x4020, 4);
WScript.echo("VarInt16 = ", VarInt16);

VarInt32 = TcClientSync.ReadInt32(0x4020, 6);
WScript.echo("VarInt32 = ", VarInt32);

VarReal32 = TcClientSync.ReadReal32(0x4020, 10);
WScript.echo("VarReal32 = ", VarReal32);


VarReal64 = TcClientSync.ReadReal64(0x4020, 14);
WScript.echo("VarReal64 = ", VarReal64);

// write to PLC
TcClientSync.WriteBool(0x4020, 0, !VarBool);
TcClientSync.WriteInt8(0x4020, 2, VarInt8 + 1);
TcClientSync.WriteInt16(0x4020, 4, VarInt16 + 1);
TcClientSync.WriteInt32(0x4020, 6, VarInt32 + 1);
TcClientSync.WriteReal32(0x4020, 10, VarReal32 + 1.1);
TcClientSync.WriteReal64(0x4020, 14, VarReal64 + 1.11);

TcClientSync.WriteVar(".PLCBoolVar", VarBool);
TcClientSync.WriteVar(".PLCSIntVar", VarInt8 +2);
TcClientSync.WriteVar(".PLCIntVar", VarInt16 + 2);
TcClientSync.WriteVar(".PLCDIntVar", VarInt32 + 2);
TcClientSync.WriteVar(".PLCRealVar", VarReal32 +2.1);
TcClientSync.WriteVar(".PLCLRealVar", VarReal64 +2.11);

// read again from PLC
VarBool = TcClientSync.ReadVar(".PLCBoolVar");
WScript.echo("VarBool = ", VarBool);

VarInt8 = TcClientSync.ReadVar(".PLCSIntVar");
WScript.echo("VarInt8 = ", VarInt8);

VarInt16 = TcClientSync.ReadVar(".PLCIntVar");
```

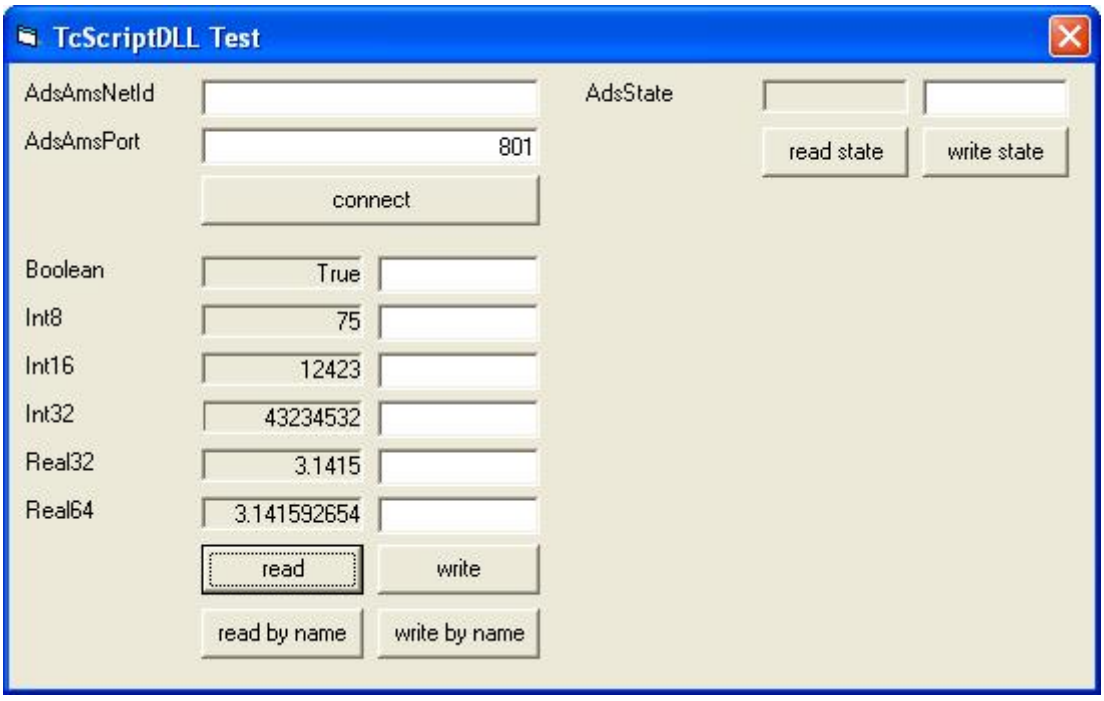
```
WScript.echo("VarInt16 = ", VarInt16);

VarInt32 = TcClientSync.ReadVar(".PLCDIntVar");
WScript.echo("VarInt32 = ", VarInt32);

VarReal32 = TcClientSync.ReadVar(".PLCRealVar");
WScript.echo("VarReal32 = ", VarReal32);

VarReal64 = TcClientSync.ReadVar(".PLCLRealVar");
WScript.echo("VarReal64 = ", VarReal64);

if (TcClientSync.ReadAdsState() == 5)
{
  WScript.echo("PLC is running!\nStop PLC.");
  TcClientSync.WriteAdsState(6);
}
```

Beispielprogramm https://infosys.beckhoff.com/content/1031/ tcscriptdll/Resources/12459716107.zip, 'Windows Scripting Host' entpacken.

4.2 Visual Basic: Lesen und Schreiben von SPS Variablen

Wie in der Einleitung schon erwähnt, ist die ADS-Script-DLL in erster Linie für Scriptsprachen entwickelt worden. Trotzdem ist an dieser Stelle ein Visual Basic Programm. Es dient legendlich dazu, die Möglichkeiten der ADS-Script-DLL besser zu demonstrieren. Für Anwendungen, die mit Visual Basic erstellt werden sollen, ist das ADS-OCX besser geeignet.

Laden Sie als erstes das SPS-Programm in die lokale SPS (Laufzeitsystem 1) auf Ihrem PC. Starten Sie anschließend das Visual Basic Programm:



Wenn Sie keine anderen Angaben machen, wird ein Kommunikationskanal zu Ihrer lokalen SPS hergestellt. Durch Eingabe einer anderen AdsAmsNetId und einer anderen Portnummer können Sie eine Verbindung zu einem anderen Laufzeitsystem auf einem anderen PC herstellen. Auf diesem Laufzeitsystem muss dann allerdings das oben genannte SPS-Programm aktiv sein. Ebenfalls muss der evtl. ausgewählte Remote Rechner im TwinCAT System Control eingetragen sein.

Sie können wahlweise die SPS-Variablen per Adresse (IndexGroup und IndexOffset), oder per Variablenname auslesen. Gleiches gilt auch für das Schreiben neuer Werte. Für den Bediener hat dieses keine Bedeutung. Es soll nur die beiden Möglichkeiten aufführen.

In der oberen rechten Ecke kann der Zustand der SPS ausgelesen und verändert werden. Eine 5 entspricht dem RUN-Zustand und eine 6 dem STOP-Zustand.

Visual Basic 6 Programm

```
Option Explicit

Public TcClientSync As TCSCRIPTLib.TcScriptSync
Public nAdsAmsPort As Integer
Public strAdsAmsNetId As String

Private Sub Form_Load()
    Set TcClientSync = CreateObject("TcScript.TcScriptSync")
    Call TcClientSync.ConnectTo("", 801)
End Sub

Private Sub Form_Terminate()
    Set TcClientSync = Nothing
End Sub

Private Sub cmdConnect_Click()
    nAdsAmsPort = CLng(txtAdsAmsPort.Text)
    if (Len(txtAdsAmsNetId.Text) > 0) Then
       strAdsAmsNetId = txtAdsAmsNetId.Text
    Else
       strAdsAmsNetId = ""
    End If
    Set TcClientSync = CreateObject("TcScript.TcScriptSync")
    Call TcClientSync.ConnectTo(strAdsAmsNetId, nAdsAmsPort)
End Sub

Private Sub cmdRead_Click()
    Dim VarBool As Variant
    Dim VarInt8 As Byte
    Dim VarInt16 As Integer
    Dim VarInt32 As Long
    Dim VarReal32 As Single
    Dim VarReal64 As Double

    VarBool = TcClientSync.ReadBool(&H4020, 0)
    lblBoolean.Caption = VarBool

    VarInt8 = TcClientSync.ReadInt8(&H4020, 2)
    lblInt8.Caption = VarInt8

    VarInt16 = TcClientSync.ReadInt16(&H4020, 4)
    lblInt16.Caption = VarInt16

    VarInt32 = TcClientSync.ReadInt32(&H4020, 6)
    lblInt32.Caption = VarInt32

    VarReal32 = TcClientSync.ReadReal32(&H4020, 10)
    lblReal32.Caption = VarReal32

    VarReal64 = TcClientSync.ReadReal64(&H4020, 14)
    lblReal64.Caption = VarReal64
End Sub

Private Sub cmdReadByName_Click()
    Dim VarBool As Variant
    Dim VarInt8 As Byte
    Dim VarInt16 As Integer
    Dim VarInt32 As Long
    Dim VarReal32 As Single
    Dim VarReal64 As Double

    VarBool = TcClientSync.ReadVar(".PLCBoolVar")
    lblBoolean.Caption = VarBool

    VarInt8 = TcClientSync.ReadVar(".PLCSIntVar")
    lblInt8.Caption = VarInt8
```

```
    VarInt16 = TcClientSync.ReadVar(".PLCIntVar")
    lblInt16.Caption = VarInt16

    VarInt32 = TcClientSync.ReadVar(".PLCDIntVar")
    lblInt32.Caption = VarInt32

    VarReal32 = TcClientSync.ReadVar(".PLCRealVar")
    lblReal32.Caption = VarReal32

    VarReal64 = TcClientSync.ReadVar(".PLCLRealVar")
    lblReal64.Caption = VarReal64
End Sub

Private Sub cmdWrite_Click()
    Dim VarBool As Boolean
    Dim VarInt8 As Byte
    Dim VarInt16 As Integer
    Dim VarInt32 As Long
    Dim VarReal32 As Single
    Dim VarReal64 As Double

    If (txtBoolean.Text <> "") Then
    VarBool = CBool(txtBoolean.Text)
    Call TcClientSync.WriteBool(&H4020, 0, VarBool)
    txtBoolean.Text = VarBool
    End If

    If (txtInt8.Text <> "") Then
    VarInt8 = CByte(txtInt8.Text)
    Call TcClientSync.WriteInt8(&H4020, 2, VarInt8)
    txtInt8.Text = VarInt8
    End If

    If (txtInt16.Text <> "") Then
    VarInt16 = CInt(txtInt16.Text)
    Call TcClientSync.WriteInt16(&H4020, 4, VarInt16)
    txtInt16.Text = VarInt16
    End If

    If (txtInt32.Text <> "") Then
    VarInt32 = CLng(txtInt32.Text)
    Call TcClientSync.WriteInt32(&H4020, 6, VarInt32)
    txtInt32.Text = VarInt32
    End If

    If (txtReal32.Text <> "") Then
    VarReal32 = CSng(txtReal32.Text)
    Call TcClientSync.WriteReal32(&H4020, 10, VarReal32)
    txtReal32.Text = VarReal32
    End If

    If (txtReal64.Text <> "") Then
    VarReal64 = CDbl(txtReal64.Text)
    Call TcClientSync.WriteReal64(&H4020, 14, VarReal64)
    txtReal64.Text = VarReal64
    End If
End Sub

Private Sub cmdWriteByName_Click()
    Dim VarBool As Boolean
    Dim VarInt8 As Byte
    Dim VarInt16 As Integer
    Dim VarInt32 As Long
    Dim VarReal32 As Single
    Dim VarReal64 As Double

    If (txtBoolean.Text <> "") Then
    VarBool = CBool(txtBoolean.Text)
    Call TcClientSync.WriteVar(".PLCBoolVar", VarBool)
    txtBoolean.Text = VarBool
    End If

    If (txtInt8.Text <> "") Then
    VarInt8 = CByte(txtInt8.Text)
    Call TcClientSync.WriteVar(".PLCSIntVar", VarInt8)
    txtInt8.Text = VarInt8
    End If

    If (txtInt16.Text <> "") Then
    VarInt16 = CInt(txtInt16.Text)
```

```
    Call TcClientSync.WriteVar(".PLCIntVar", VarInt16)
    txtInt16.Text = VarInt16
    End If

    If (txtInt32.Text <> "") Then
    VarInt32 = CLng(txtInt32.Text)
    Call TcClientSync.WriteVar(".PLCDIntVar", VarInt32)
    txtInt32.Text = VarInt32
    End If

    If (txtReal32.Text <> "") Then
    VarReal32 = CSng(txtReal32.Text)
    Call TcClientSync.WriteVar(".PLCRealVar", VarReal32)
    txtReal32.Text = VarReal32
    End If

    If (txtReal64.Text <> "") Then
    VarReal64 = CDbl(txtReal64.Text)
    Call TcClientSync.WriteVar(".PLCLRealVar", VarReal64)
    txtReal64.Text = VarReal64
    End If
End Sub

Private Sub cmdReadState_Click()
    lblAdsState.Caption = TcClientSync.ReadAdsState()
End Sub

Private Sub cmdWriteState_Click()
    If (txtAdsState.Text <> "") Then
    If (txtAdsState.Text <> TcClientSync.ReadAdsState()) Then
        Call TcClientSync.WriteAdsState(txtAdsState.Text)
    End If
    End If
End Sub
```

SPS Programm

```
VAR_GLOBAL
    PLCBoolVar      AT %MX0.0: BOOL := TRUE;
    PLCSIntVar      AT %MB2 : USINT := 75;
    PLCIntVar       AT %MB4 : UINT := 12423;
    PLCDIntVar      AT %MB6 : UDINT := 43234532;
    PLCRealVar      AT %MB10    : REAL := 3.1415;
    PLCLRealVar     AT %MB14    : LREAL := 3.141592654;
END_VAR
```

Sprache / IDE	Beispielprogramm auspacken
Visual Basic 6	https://infosys.beckhoff.com/content/1031/ tcscriptdll/ Resources/12459720971.zip

4.3 Active Server Pages

Active Server Pages (ASP) bieten die Möglichkeit HTML-Seiten dynamisch, also in Abhängigkeiten von bestimmten Ereignissen, auf dem HTTP-Server zu erzeugen. Erreicht wird dieses durch Einbetten von Scriptsegmenten in den HTML-Code. Die Scripte werden hierbei vom Web-Server ausgeführt. Für den Zugriff auf TwinCAT wird hierbei die TcScript.dll eingesetzt. Das folgende Beispiel benutzt den Internet Information Server (IIS) bzw. den Personal Web Server (PWS) von Microsoft.
Das Erzeugen einer TcScript.dll-Instanz kann am einfachsten in der global.asa durchgeführt werden. Die erzeugte Instanz ist innerhalb der Web-Applikation für alle ASP-Seiten gültig.

global.asa:

```
<OBJECT RUNAT="Server" SCOPE="Application" ID="TcPLC" PROGID="TcScript.TcScriptSync"> </OBJECT>
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">

Sub Application_OnStart()
  Call TcPLC.ConnectTo("", 801)
```

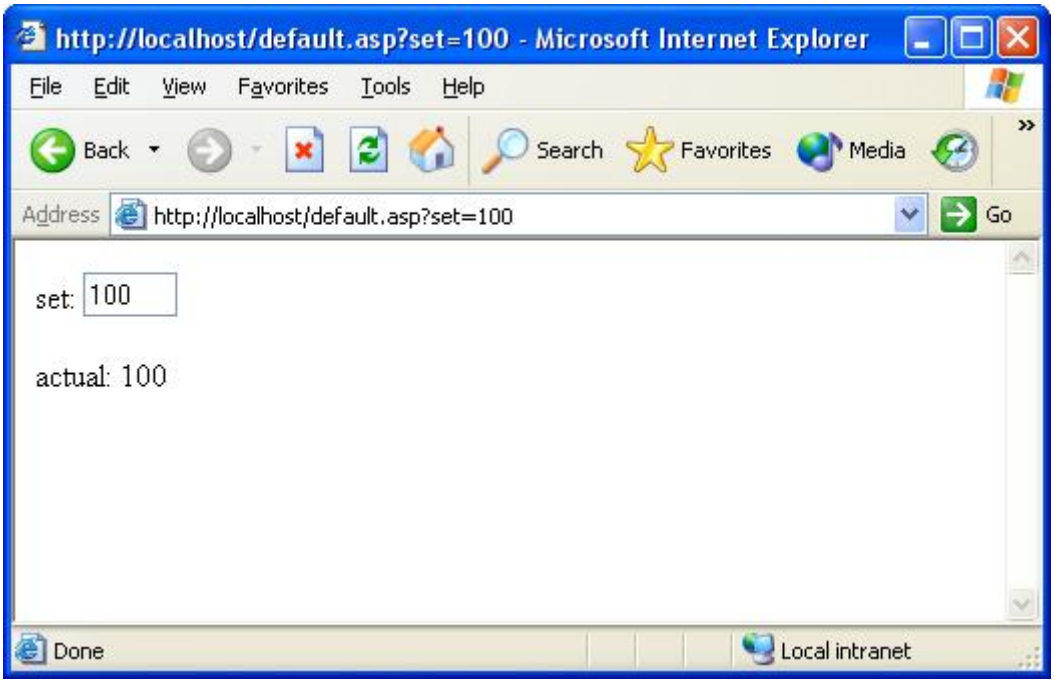
```
End Sub

</SCRIPT>
```

An der URL einer ASP-Seite können Parameter angehängt werden. Diese Parameter werden durch ein '?' von der URL getrennt und haben den Aufbau *VarName=VarWert.* Am Anfang der ASP-Seite wird überprüft ob der Parameter *set* übergeben wurde und ob der Wert in eine Zahl konvertiert werden kann. Ist dieses der Fall, wird der Parameter in die VBScript-Variable *intSet* kopiert und anschließend in die globale SPS-Variable *PLCVarInt* übertragen. Am Ende des Scripts wird die SPS-Variable ausgelesen und in die VBScript-Variable *intActual* geschrieben. Das Auslesen der SPS-Variable wird auch dann durchgeführt, wenn kein Parameter an die ASP-Seite übergeben wurde.
Innerhalb des HTML-Bereichs wird über das Response-Objekt (stellt ASP zur Verfügung) der Wert der VBScript-Variable *intActual* in den HTML-Code eingesetzt und zum Client übertragen. D.h. jedes Mal, wenn die ASP-Seite aufgerufen wird (mit oder ohne Parameter), wird der aktuelle Wert der SPS-Variablen im Explorer angezeigt.
Um Eingaben vom Bediener entgegen zu nehmen, werden innerhalb von HTML Formulare benutzt. Die Eingaben werden dabei an eine bestimmten Seite als Parameter übergeben. Bei unserem Beispiel wird die gleiche Seite (default.asp) wieder aufgerufen. Als Parameter wird *set* übergeben (Name des Eingabefeldes), mit dem entsprechenden Wert, den der Bediener eingeben hat. Da jetzt die Seite mit dem Parameter *set* erneut aufgerufen wird, wird der Wert hinter dem *set*-Parameter in die SPS geschrieben.

default.asp:

```
<%@ Language=VBScript %>
<%Option Explicit
Dim intActual, intSet
Application.Lock
If Not (IsEmpty(Request.Querystring("set"))) Then
  If (IsNumeric(Request.Querystring("set"))) Then
    intSet = cint(Request.Querystring("set"))
    call TcPLC.WriteVar(".PLCVarInt", intSet)
  End if
End If
intActual = TcPLC.ReadVar(".PLCVarInt")
Application.Unlock%>

<html>
<head><meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"></head>
<body>
<form method="get" action="default.asp" name="tempSet">set: <input name="set" size="4" value="<% Res
ponse.Write intSet %>"></form>
actual: <% Response.Write intActual %>
</body>
</html>
```

Darstellung im Internet Explorer:



Der Client (Internet Explorer) bekommt hierbei reines HTML zugeschickt. Dadurch können die Seiten auch von Rechnern aufgerufen werden, auf denen kein TwinCAT installiert ist (z.B. Handheld PC, oder auch Smart-Phones). Der Zugriff auf TwinCAT wird hierbei ausschließlich von Web-Server durchgeführt.

HTML-Seite die zum Client (Internet Explorer) geschickt wurde:

```
<html>
<head><meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"></head>
<body>
<form method="get" action="default.asp" name="tempSet">set: <input name="set" size="4" value="100"><
/form>
actual: 100
</body>
</html>
```

Beispielprogramm https://infosys.beckhoff.com/content/1031/ tcscriptdll/Resources/12459711755.zip, 'Active Server Pages' entpacken.

4.4 Active Server Pages für Windows CE

Dieses Beispiel ist eine Sonderform von Beispiel 3 [▶ 33]. Machen Sie sich daher bitte erst mit der in Beispiel 3 [▶ 33] beschriebenen Methodik vertraut.

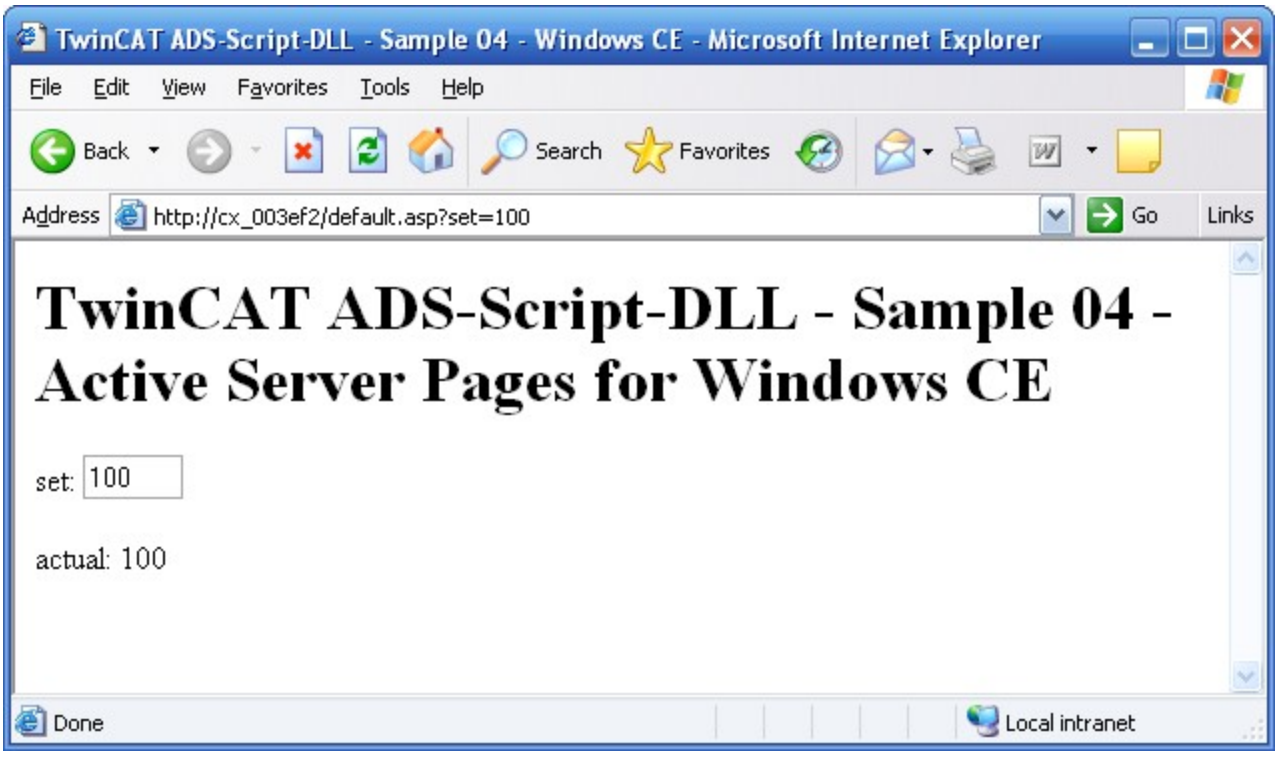
Windows CE kann als Web-Server für ASP-Seiten dienen (indem die entsprechenden Dateien in das "WWW"-Verzeichnis kopiert werden). Die Funktionalität hiervon ist jedoch eingeschränkt, was sich darin äußert, dass auf Windows CE die "*global.asa*"-Datei nicht verarbeitet werden kann. Somit ist es unter Anderem nicht möglich globale Variablen und Objekte zu erstellen; sie müssen bei jedem Seitenaufruf neu angelegt werden, indem (wie nachfolgend dargestellt) sie direkt im Script instanziiert werden.

default.asp:

```
<%@ language=JScript %>
<%
   TcClient = new ActiveXObject("TcScript.TcScriptSync");
   TcClient.ConnectTo("", 801);

   var nIntSet = 0;
```

```
   var nIntActual = 0;
   var sSet = Request.QueryString("set");

   if ((sSet != null) && (sSet.length != 0))
   {
      var bSetIsNumber = true;

      for (var i = 0; ((i < sSet.length) && (bSetIsNumber == true)); i++)
      {
     var sChar = sSet.charAt(i);
     if (ValidChars.indexOf("0123456789.") == -1)
     {
        bSetIsNumber = false;
     }
      }

      if (bSetIsNumber == true)
      {
     nIntSet = parseInt(sSet);
     if(nIntSet >= 0)
     {
        TcClient.WriteVar(".PLCVarInt", nIntSet);
     }
      }
   }

   nIntActual = TcClient.ReadVar(".PLCVarInt");
%>

<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
      <title>TwinCAT ADS-Script-DLL - Sample 04 - Windows CE</title>
   </head>

   <body>
      <h1>TwinCAT ADS-Script-DLL - Sample 04 - Active Server Pages for Windows CE</h1>
      <form method="get" action="default.asp" name="tempSet">
     set: <input name="set" size="4" value="<% Response.Write( nIntSet ); %>">
      </form>
      actual: <% Response.Write( nIntActual ); %>
   </body>
</html>
```

Darstellung im Internet Explorer:



HTML-Seite die zum Client (Internet Explorer) geschickt wurde:

```
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
      <title>TwinCAT ADS-Script-DLL - Sample 04 - Windows CE</title>
   </head>

   <body>
      <h1>TwinCAT ADS-Script-DLL - Sample 04 - Active Server Pages for Windows CE</h1>
      <form method="get" action="Sample04.asp" name="tempSet">
     set: <input name="set" size="4" value="3">
      </form>
      actual: 3
   </body>
</html>
```

Beispielprogramm https://infosys.beckhoff.com/content/1031/ tcscriptdll/Resources/12459724811.zip 'Active Server Pages für Windows CE' entpacken.

5 ADS Return Codes

Gruppierung der Fehlercodes:
Globale Fehlercodes: 0x0000 [▶ 38]... (0x9811_0000 ...)
Router Fehlercodes: 0x0500 [▶ 38]... (0x9811_0500 ...)
Allgemeine ADS Fehler: 0x0700 [▶ 39]... (0x9811_0700 ...)
RTime Fehlercodes: 0x1000 [▶ 40]... (0x9811_1000 ...)

Globale Fehlercodes

Hex	Dec	HRESULT	Name	Beschreibung
0x0	0	0x98110000	ERR_NOERROR	Kein Fehler.
0x1	1	0x98110001	ERR_INTERNAL	Interner Fehler.
0x2	2	0x98110002	ERR_NORTIME	Keine Echtzeit.
0x3	3	0x98110003	ERR_ALLOCLOCKEDMEM	Zuweisung gesperrt - Speicherfehler.
0x4	4	0x98110004	ERR_INSERTMAILBOX	Postfach voll – Es konnte die ADS Nachricht nicht versendet werden. Reduzieren der Anzahl der ADS Nachrichten pro Zyklus bringt Abhilfe.
0x5	5	0x98110005	ERR_WRONGRECEIVEHMSG	Falsches HMSG.
0x6	6	0x98110006	ERR_TARGETPORTNOTFOUND	Ziel-Port nicht gefunden – ADS Server ist nicht gestartet oder erreichbar.
0x7	7	0x98110007	ERR_TARGETMACHINENOTFOUND	Zielrechner nicht gefunden – AMS Route wurde nicht gefunden.
0x8	8	0x98110008	ERR_UNKNOWNCMDID	Unbekannte Befehl-ID.
0x9	9	0x98110009	ERR_BADTASKID	Ungültige Task-ID.
0xA	10	0x9811000A	ERR_NOIO	Kein IO.
0xB	11	0x9811000B	ERR_UNKNOWNAMSCMD	Unbekannter AMS-Befehl.
0xC	12	0x9811000C	ERR_WIN32ERROR	Win32 Fehler.
0xD	13	0x9811000D	ERR_PORTNOTCONNECTED	Port nicht verbunden.
0xE	14	0x9811000E	ERR_INVALIDAMSLENGTH	Ungültige AMS-Länge.
0xF	15	0x9811000F	ERR_INVALIDAMSNETID	Ungültige AMS Net ID.
0x10	16	0x98110010	ERR_LOWINSTLEVEL	Installations-Level ist zu niedrig –TwinCAT 2 Lizenzfehler.
0x11	17	0x98110011	ERR_NODEBUGINTAVAILABLE	Kein Debugging verfügbar.
0x12	18	0x98110012	ERR_PORTDISABLED	Port deaktiviert – TwinCAT System Service nicht gestartet.
0x13	19	0x98110013	ERR_PORTALREADYCONNECTED	Port bereits verbunden.
0x14	20	0x98110014	ERR_AMSSYNC_W32ERROR	AMS Sync Win32 Fehler.
0x15	21	0x98110015	ERR_AMSSYNC_TIMEOUT	AMS Sync Timeout.
0x16	22	0x98110016	ERR_AMSSYNC_AMSERROR	AMS Sync Fehler.
0x17	23	0x98110017	ERR_AMSSYNC_NOINDEXINMAP	Keine Index-Map für AMS Sync vorhanden.
0x18	24	0x98110018	ERR_INVALIDAMSPORT	Ungültiger AMS-Port.
0x19	25	0x98110019	ERR_NOMEMORY	Kein Speicher.
0x1A	26	0x9811001A	ERR_TCPSEND	TCP Sendefehler.
0x1B	27	0x9811001B	ERR_HOSTUNREACHABLE	Host nicht erreichbar.
0x1C	28	0x9811001C	ERR_INVALIDAMSFRAGMENT	Ungültiges AMS Fragment.
0x1D	29	0x9811001D	ERR_TLSSEND	TLS Sendefehler – Secure ADS Verbindung fehlgeschlagen.
0x1E	30	0x9811001E	ERR_ACCESSDENIED	Zugriff Verweigert – Secure ADS Zugriff verweigert.

Router Fehlercodes

Hex	Dec	HRESULT	Name	Beschreibung
0x500	1280	0x98110500	ROUTERERR_NOLOCKEDMEMORY	Lockierter Speicher kann nicht zugewiesen werden.
0x501	1281	0x98110501	ROUTERERR_RESIZEMEMORY	Die Größe des Routerspeichers konnte nicht geändert werden.
0x502	1282	0x98110502	ROUTERERR_MAILBOXFULL	Das Postfach hat die maximale Anzahl der möglichen Meldungen erreicht.
0x503	1283	0x98110503	ROUTERERR_DEBUGBOXFULL	Das Debug Postfach hat die maximale Anzahl der möglichen Meldungen erreicht.

Hex	Dec	HRESULT	Name	Beschreibung
0x504	1284	0x98110504	ROUTERERR_UNKNOWNPORTTYPE	Der Porttyp ist unbekannt.
0x505	1285	0x98110505	ROUTERERR_NOTINITIALIZED	Router ist nicht initialisiert.
0x506	1286	0x98110506	ROUTERERR_PORTALREADYINUSE	Die Portnummer ist bereits vergeben.
0x507	1287	0x98110507	ROUTERERR_NOTREGISTERED	Der Port ist nicht registriert.
0x508	1288	0x98110508	ROUTERERR_NOMOREQUEUES	Die maximale Portanzahl ist erreicht.
0x509	1289	0x98110509	ROUTERERR_INVALIDPORT	Der Port ist ungültig.
0x50A	1290	0x9811050A	ROUTERERR_NOTACTIVATED	Der Router ist nicht aktiv.
0x50B	1291	0x9811050B	ROUTERERR_FRAGMENTBOXFULL	Das Postfach hat die maximale Anzahl für fragmentierte Nachrichten erreicht.
0x50C	1292	0x9811050C	ROUTERERR_FRAGMENTTIMEOUT	Fragment Timeout aufgetreten.
0x50D	1293	0x9811050D	ROUTERERR_TOBEREMOVED	Port wird entfernt.

Allgemeine ADS Fehlercodes

Hex	Dec	HRESULT	Name	Beschreibung
0x700	1792	0x98110700	ADSERR_DEVICE_ERROR	Allgemeiner Gerätefehler.
0x701	1793	0x98110701	ADSERR_DEVICE_SRVNOTSUPP	Service wird vom Server nicht unterstützt.
0x702	1794	0x98110702	ADSERR_DEVICE_INVALIDGRP	Ungültige Index-Gruppe.
0x703	1795	0x98110703	ADSERR_DEVICE_INVALIDOFFSET	Ungültiger Index-Offset.
0x704	1796	0x98110704	ADSERR_DEVICE_INVALIDACCESS	Lesen oder Schreiben nicht gestattet.
0x705	1797	0x98110705	ADSERR_DEVICE_INVALIDSIZE	Parametergröße nicht korrekt.
0x706	1798	0x98110706	ADSERR_DEVICE_INVALIDDATA	Ungültige Daten-Werte.
0x707	1799	0x98110707	ADSERR_DEVICE_NOTREADY	Gerät nicht betriebsbereit.
0x708	1800	0x98110708	ADSERR_DEVICE_BUSY	Gerät beschäftigt.
0x709	1801	0x98110709	ADSERR_DEVICE_INVALIDCONTEXT	Ungültiger Kontext vom Betriebssystem - Kann durch Verwendung von ADS Bausteinen in unterschiedlichen Tasks auftreten. Abhilfe kann die Multitasking-Syncronisation in der SPS geben.
0x70A	1802	0x9811070A	ADSERR_DEVICE_NOMEMORY	Nicht genügend Speicher.
0x70B	1803	0x9811070B	ADSERR_DEVICE_INVALIDPARM	Ungültige Parameter-Werte.
0x70C	1804	0x9811070C	ADSERR_DEVICE_NOTFOUND	Nicht gefunden (Dateien,...).
0x70D	1805	0x9811070D	ADSERR_DEVICE_SYNTAX	Syntax-Fehler in Datei oder Befehl.
0x70E	1806	0x9811070E	ADSERR_DEVICE_INCOMPATIBLE	Objekte stimmen nicht überein.
0x70F	1807	0x9811070F	ADSERR_DEVICE_EXISTS	Objekt ist bereits vorhanden.
0x710	1808	0x98110710	ADSERR_DEVICE_SYMBOLNOTFOUND	Symbol nicht gefunden.
0x711	1809	0x98110711	ADSERR_DEVICE_SYMBOLVERSIONINVALID	Symbol-Version ungültig – Kann durch einen Online-Change auftreten. Erzeuge einen neuen Handle.
0x712	1810	0x98110712	ADSERR_DEVICE_INVALIDSTATE	Gerät (Server) ist im ungültigen Zustand.
0x713	1811	0x98110713	ADSERR_DEVICE_TRANSMODENOTSUPP	AdsTransMode nicht unterstützt.
0x714	1812	0x98110714	ADSERR_DEVICE_NOTIFYHNDINVALID	Notification Handle ist ungültig.
0x715	1813	0x98110715	ADSERR_DEVICE_CLIENTUNKNOWN	Notification-Client nicht registriert.
0x716	1814	0x98110716	ADSERR_DEVICE_NOMOREHDLS	Keine weiteren Handles verfügbar.
0x717	1815	0x98110717	ADSERR_DEVICE_INVALIDWATCHSIZE	Größe der Notification zu groß.
0x718	1816	0x98110718	ADSERR_DEVICE_NOTINIT	Gerät nicht initialisiert.
0x719	1817	0x98110719	ADSERR_DEVICE_TIMEOUT	Gerät hat einen Timeout.
0x71A	1818	0x9811071A	ADSERR_DEVICE_NOINTERFACE	Interface Abfrage fehlgeschlagen.
0x71B	1819	0x9811071B	ADSERR_DEVICE_INVALIDINTERFACE	Falsches Interface angefordert.
0x71C	1820	0x9811071C	ADSERR_DEVICE_INVALIDCLSID	Class-ID ist ungültig.
0x71D	1821	0x9811071D	ADSERR_DEVICE_INVALIDOBJID	Object-ID ist ungültig.
0x71E	1822	0x9811071E	ADSERR_DEVICE_PENDING	Anforderung steht aus.
0x71F	1823	0x9811071F	ADSERR_DEVICE_ABORTED	Anforderung wird abgebrochen.
0x720	1824	0x98110720	ADSERR_DEVICE_WARNING	Signal-Warnung.
0x721	1825	0x98110721	ADSERR_DEVICE_INVALIDARRAYIDX	Ungültiger Array-Index.
0x722	1826	0x98110722	ADSERR_DEVICE_SYMBOLNOTACTIVE	Symbol nicht aktiv.
0x723	1827	0x98110723	ADSERR_DEVICE_ACCESSDENIED	Zugriff verweigert.
0x724	1828	0x98110724	ADSERR_DEVICE_LICENSENOTFOUND	Fehlende Lizenz.
0x725	1829	0x98110725	ADSERR_DEVICE_LICENSEEXPIRED	Lizenz abgelaufen.
0x726	1830	0x98110726	ADSERR_DEVICE_LICENSEEXCEEDED	Lizenz überschritten.
0x727	1831	0x98110727	ADSERR_DEVICE_LICENSEINVALID	Lizenz ungültig.

Hex	Dec	HRESULT	Name	Beschreibung
0x728	1832	0x98110728	ADSERR_DEVICE_LICENSESYSTEMID	Lizenzproblem: System-ID ist ungültig.
0x729	1833	0x98110729	ADSERR_DEVICE_LICENSENOTIMELIMIT	Lizenz nicht zeitlich begrenzt.
0x72A	1834	0x9811072A	ADSERR_DEVICE_LICENSEFUTUREISSUE	Lizenzproblem: Zeitpunkt in der Zukunft.
0x72B	1835	0x9811072B	ADSERR_DEVICE_LICENSETIMETOLONG	Lizenz-Zeitraum zu lang.
0x72C	1836	0x9811072C	ADSERR_DEVICE_EXCEPTION	Exception beim Systemstart.
0x72D	1837	0x9811072D	ADSERR_DEVICE_LICENSEDUPLICATED	Lizenz-Datei zweimal gelesen.
0x72E	1838	0x9811072E	ADSERR_DEVICE_SIGNATUREINVALID	Ungültige Signatur.
0x72F	1839	0x9811072F	ADSERR_DEVICE_CERTIFICATEINVALID	Zertifikat ungültig.
0x730	1840	0x98110730	ADSERR_DEVICE_LICENSEOEMNOTFOUND	Public Key vom OEM nicht bekannt.
0x731	1841	0x98110731	ADSERR_DEVICE_LICENSERESTRICTED	Lizenz nicht gültig für diese System.ID.
0x732	1842	0x98110732	ADSERR_DEVICE_LICENSEDEMODENIED	Demo-Lizenz untersagt.
0x733	1843	0x98110733	ADSERR_DEVICE_INVALIDFNCID	Funktions-ID ungültig.
0x734	1844	0x98110734	ADSERR_DEVICE_OUTOFRANGE	Außerhalb des gültigen Bereiches.
0x735	1845	0x98110735	ADSERR_DEVICE_INVALIDALIGNMENT	Ungültiges Alignment.
0x736	1846	0x98110736	ADSERR_DEVICE_LICENSEPLATFORM	Ungültiger Plattform Level.
0x737	1847	0x98110737	ADSERR_DEVICE_FORWARD_PL	Kontext – Weiterleitung zum Passiv-Level.
0x738	1848	0x98110738	ADSERR_DEVICE_FORWARD_DL	Kontext – Weiterleitung zum Dispatch-Level.
0x739	1849	0x98110739	ADSERR_DEVICE_FORWARD_RT	Kontext – Weiterleitung zur Echtzeit.
0x740	1856	0x98110740	ADSERR_CLIENT_ERROR	Clientfehler.
0x741	1857	0x98110741	ADSERR_CLIENT_INVALIDPARM	Dienst enthält einen ungültigen Parameter.
0x742	1858	0x98110742	ADSERR_CLIENT_LISTEMPTY	Polling-Liste ist leer.
0x743	1859	0x98110743	ADSERR_CLIENT_VARUSED	Var-Verbindung bereits im Einsatz.
0x744	1860	0x98110744	ADSERR_CLIENT_DUPLINVOKEID	Die aufgerufene ID ist bereits in Benutzung.
0x745	1861	0x98110745	ADSERR_CLIENT_SYNCTIMEOUT	Timeout ist aufgetreten – Die Gegenstelle antwortet nicht im vorgegebenen ADS Timeout. Die Routeneinstellung der Gegenstelle kann falsch konfiguriert sein.
0x746	1862	0x98110746	ADSERR_CLIENT_W32ERROR	Fehler im Win32 Subsystem.
0x747	1863	0x98110747	ADSERR_CLIENT_TIMEOUTINVALID	Ungültiger Client Timeout-Wert.
0x748	1864	0x98110748	ADSERR_CLIENT_PORTNOTOPEN	Port nicht geöffnet.
0x749	1865	0x98110749	ADSERR_CLIENT_NOAMSADDR	Keine AMS Adresse.
0x750	1872	0x98110750	ADSERR_CLIENT_SYNCINTERNAL	Interner Fehler in Ads-Sync.
0x751	1873	0x98110751	ADSERR_CLIENT_ADDHASH	Überlauf der Hash-Tabelle.
0x752	1874	0x98110752	ADSERR_CLIENT_REMOVEHASH	Schlüssel in der Tabelle nicht gefunden.
0x753	1875	0x98110753	ADSERR_CLIENT_NOMORESYM	Keine Symbole im Cache.
0x754	1876	0x98110754	ADSERR_CLIENT_SYNCRESINVALID	Ungültige Antwort erhalten.
0x755	1877	0x98110755	ADSERR_CLIENT_SYNCPORTLOCKED	Sync Port ist verriegelt.

RTime Fehlercodes

Hex	Dec	HRESULT	Name	Beschreibung
0x1000	4096	0x98111000	RTERR_INTERNAL	Interner Fehler im Echtzeit-System.
0x1001	4097	0x98111001	RTERR_BADTIMERPERIODS	Timer-Wert nicht gültig.
0x1002	4098	0x98111002	RTERR_INVALIDTASKPTR	Task-Pointer hat den ungültigen Wert 0 (null).
0x1003	4099	0x98111003	RTERR_INVALIDSTACKPTR	Stack-Pointer hat den ungültigen Wert 0 (null).
0x1004	4100	0x98111004	RTERR_PRIOEXISTS	Die Request Task Priority ist bereits vergeben.
0x1005	4101	0x98111005	RTERR_NOMORETCB	Kein freier TCB (Task Control Block) verfügbar. Maximale Anzahl von TCBs beträgt 64.
0x1006	4102	0x98111006	RTERR_NOMORESEMAS	Keine freien Semaphoren zur Verfügung. Maximale Anzahl der Semaphoren beträgt 64.
0x1007	4103	0x98111007	RTERR_NOMOREQUEUES	Kein freier Platz in der Warteschlange zur Verfügung. Maximale Anzahl der Plätze in der Warteschlange beträgt 64.
0x100D	4109	0x9811100D	RTERR_EXTIRQALREADYDEF	Ein externer Synchronisations-Interrupt wird bereits angewandt.
0x100E	4110	0x9811100E	RTERR_EXTIRQNOTDEF	Kein externer Sync-Interrupt angewandt.
0x100F	4111	0x9811100F	RTERR_EXTIRQINSTALLFAILED	Anwendung des externen Synchronisierungs-Interrupts ist fehlgeschlagen.
0x1010	4112	0x98111010	RTERR_IRQLNOTLESSOREQUAL	Aufruf einer Service-Funktion im falschen Kontext
0x1017	4119	0x98111017	RTERR_VMXNOTSUPPORTED	Intel VT-x Erweiterung wird nicht unterstützt.
0x1018	4120	0x98111018	RTERR_VMXDISABLED	Intel VT-x Erweiterung ist nicht aktiviert im BIOS.

Hex	Dec	HRESULT	Name	Beschreibung
0x1019	4121	0x98111019	RTERR_VMXCONTROLSMISSING	Fehlende Funktion in Intel VT-x Erweiterung.
0x101A	4122	0x9811101A	RTERR_VMXENABLEFAILS	Aktivieren von Intel VT-x schlägt fehl.

Spezifische positive HRESULT Return Codes:

HRESULT	Name	Beschreibung
0x0000_0000	S_OK	Kein Fehler.
0x0000_0001	S_FALSE	Kein Fehler. Bsp.: erfolgreiche Abarbeitung, bei der jedoch ein negatives oder unvollständiges Ergebnis erzielt wurde.
0x0000_0203	S_PENDING	Kein Fehler. Bsp.: erfolgreiche Abarbeitung, bei der jedoch noch kein Ergebnis vorliegt.
0x0000_0256	S_WATCHDOG_TIMEOUT	Kein Fehler. Bsp.: erfolgreiche Abarbeitung, bei der jedoch eine Zeitüberschreitung eintrat.

TCP Winsock-Fehlercodes

Hex	Dec	Name	Beschreibung
0x274C	10060	WSAETIMEDOUT	Verbindungs Timeout aufgetreten - Fehler beim Herstellen der Verbindung, da die Gegenstelle nach einer bestimmten Zeitspanne nicht ordnungsgemäß reagiert hat, oder die hergestellte Verbindung konnte nicht aufrecht erhalten werden, da der verbundene Host nicht reagiert hat.
0x274D	10061	WSAECONNREFUSED	Verbindung abgelehnt - Es konnte keine Verbindung hergestellt werden, da der Zielcomputer dies explizit abgelehnt hat. Dieser Fehler resultiert normalerweise aus dem Versuch, eine Verbindung mit einem Dienst herzustellen, der auf dem fremden Host inaktiv ist—das heißt, einem Dienst, für den keine Serveranwendung ausgeführt wird.
0x2751	10065	WSAEHOSTUNREACH	Keine Route zum Host - Ein Socketvorgang bezog sich auf einen nicht verfügbaren Host.
Weitere Winsock-Fehlercodes: Win32-Fehlercodes			

Mehr Informationen:
www.beckhoff.de/automation

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Deutschland
Telefon: +49 5246 9630
info@beckhoff.de
www.beckhoff.de