IP-S7-Link .Net Advanced COM

© by TIS

The IP-S7-Link .Net Advanced COM is a library which uses the software driver framework IP-S7-Link .Net Advanced to communicate and exchange data with programmable logic controllers (PLC) in the industrial automation space and in other industries. The library provides an application programming interface to interact with PLC devices using the internet protocol (IP) stack for Simatic S7 controllers. This interface has been implemented using the Component Object Model (COM) from Microsoft. Through that the library can be used within Visual Basic 6 (VB6), Visual Basic Script (VBScript) and Visual Basic for Applications (VBA). There VBA does include all Microsoft Office products supporting VBA Macro programming like Microsoft Excel.

The documentation for the library includes a class library reference, conceptual overview, step-by-step procedures, and information about samples. To locate the information that interests you, see the following list of main topic areas.

Overview Section

IP-S7-Link .Net Advanced COM Essentials
Provides a quick view into the technical terms relating to communicate with PLC devices.

Getting started with the IP-S7-Link .Net Advanced COM
Provides a comprehensive overview of the IP-S7-Link .Net Advanced COM library and links to additional resources.

Getting started with the IP-S7-Link .Net Advanced COM in Microsoft Excel
Provides step by step instructions how to take use of the IP-S7-Link .Net Advanced COM library in Microsoft Excel using Visual Basic for Applications (VBA) to develop VBA Macros.

IP-S7-Link .Net Advanced COM Code Snippets
Provides a collection of instant scenario based code snippets to directly jump into the solution for the most common requirements during development.

IP-S7-Link .Net Advanced COM Frequently Asked Questions (FAQs)
Provides all answers to all frequently asked questions which can be typically solved by the same procedure.

IP-S7-Link .Net Advanced COM Essentials

© by TIS

What is the IP-S7-Link .Net Advanced COM?

The IP-S7-Link .Net Advanced COM is a library which uses the software driver framework IP-S7-Link .Net Advanced which does provide a whole implementation of the internet protocol (IP) stack used to communicate and exchange data with programmable logic controllers (PLC). While the framework does realize a fluently transition of PC specific data to PLC specific data formats and vice versa. And while the implemented communication stack does realize complex optimization algorithms to reduce the amount and costs of interactions with the PLC device, the library does provide a nearly one-to-one interface from COM to the framework. This is done using a fully object-orientated API which does also take care about a fluently transition of COM specific data to PLC specific data formats and vice versa. Through that it does package the whole logic into fluently class interfaces which target common automation scenarios in COM based applications and not any knowledge about the stack behind the driver is required.

Verwandte Themen

Title	**Description**
SIMATIC controllers from Siemens	A short overview of the Automation Systems provided by the Siemens AG.
Technical documentation SIMATIC Controller	Provides a set of documents regarding the different automation solutions provided by the Siemens AG.
SIMATIC Programming with STEP 7	The whole documentation to develop using STEP 7.

Getting started with the IP-S7-Link .Net Advanced COM

The IP-S7-Link .Net Advanced COM is a library which uses the software driver framework IP-S7-Link .Net Advanced which does implement the whole internet protocol (IP) stack for Simatic S7 controllers. While the library does provide simple read / write methods it does also target different advanced scenarios. Using this library does not only bring the possibility to bind programmable logic controller (PLC) addresses to your COM application. It does also provide mechanisms to link variables or whole structs to one or more PLC addresses of your PLC device and read / write them at once.

This section of the IP-S7-Link .Net Advanced COM documentation provides information about basic application development tasks.

Operand

Name	**Abbreviation (Siemens, DE)**	**Abbreviation(IEC)**
Input	E	I
Output	A	Q
Flag	M	M
Peripherals	P	P
Counter	Z	C
Data Block	DB	DB
Timer	T	16

Data types

Name	**Abbreviation**	**Bit size**	**Range**	**Description**	**Array**
BOOL	X	1	0 to 1	single bit representing true (1) or false (0)	x
BYTE	B	8	0 to 255	unsigned 8-bit	x
WORD	W	16	0 to 65.535	unsigned 16-bit (Word)	x
DWORD	D	32	0 to 2^{32} -1	unsigned 32-bit (Double Word)	x
CHAR	B	8	A+00 to A+ff	ASCII-Code unsigned 8-bit character	x
INT	W	16	-32.768 to 32.767	signed 16-bit integer	x
DINT	D	32	-2^{31} to 2^{31}-1	signed 32-bit integer (Double Word)	x
REAL	D	32	+-1.5e-45 to +-3.4e38	IEEE754 32-bit single precision floating point number	x
S5TIME	W	16	00.00:00:00.100 to 00.02:46:30.000	binary coded decimal (BCD) number representing a time span	
TIME	D	32	00.00:00:00.000 to 24.20:31:23.647	signed 16-bit integer representing a time span in milliseconds	

Name	Abbreviation	Bit size	Range	Description	Array
TIME_OF_DAY	D	32	00.00:00:00.000 to 00.23:59:59.999	unsigned 16-bit integer representing a time span in milliseconds	
DATE	W	16	01.01.1990 to 31.12.2168	unsigned 16-bit integer representing a date in days	
DATE_AND_TIME	D	64	00:00:00.000 01.01.1990 to 23:59:59.999 31.12.2089	binary coded decimal (BCD) number representing a date and time	
S7String	B	any	A+00 to A+ff	ASCII-Code, max. 254 Bytes	

The variables are composed of operand and data type. Examples:

Examples	Data type	Example Siemens	Example IEC
Input Byte 1, Bit 0	BOOL	E 1.0	I 1.0
Output Byte 1, Bit 7	BOOL	A 1.7	Q 1.7
Flag Byte 10, Bit 1	BOOL	M 10.1	M 10.1
Data Block 1, Byte 1, Bit 0	BOOL	DB1.DBX 1.0	DB1.DBX 1.0
Input Byte 1	BYTE	EB 1	IB 1
Output Byte 10	BYTE	AB 10	QB 10
Flag Byte 100	BYTE	MB 100	MB 100
Peripherals Input Byte 0	BYTE	PEB 0	PIB 0
Peripherals Output Byte 1	BYTE	PAB 1	PQB 1
Data Block 1, Byte 1	BYTE	DB1.DBB 1	DB1.DBB 1

Data Block 1, Data Block 1 Typ bool, Address 1.0 → DB1.DBX 1.0
Data Block 1, Data Block Typ Byte, Address 1 → DB1.DBB 1
Peripherals Input, Typ DWORD, Address 0 → PED 0

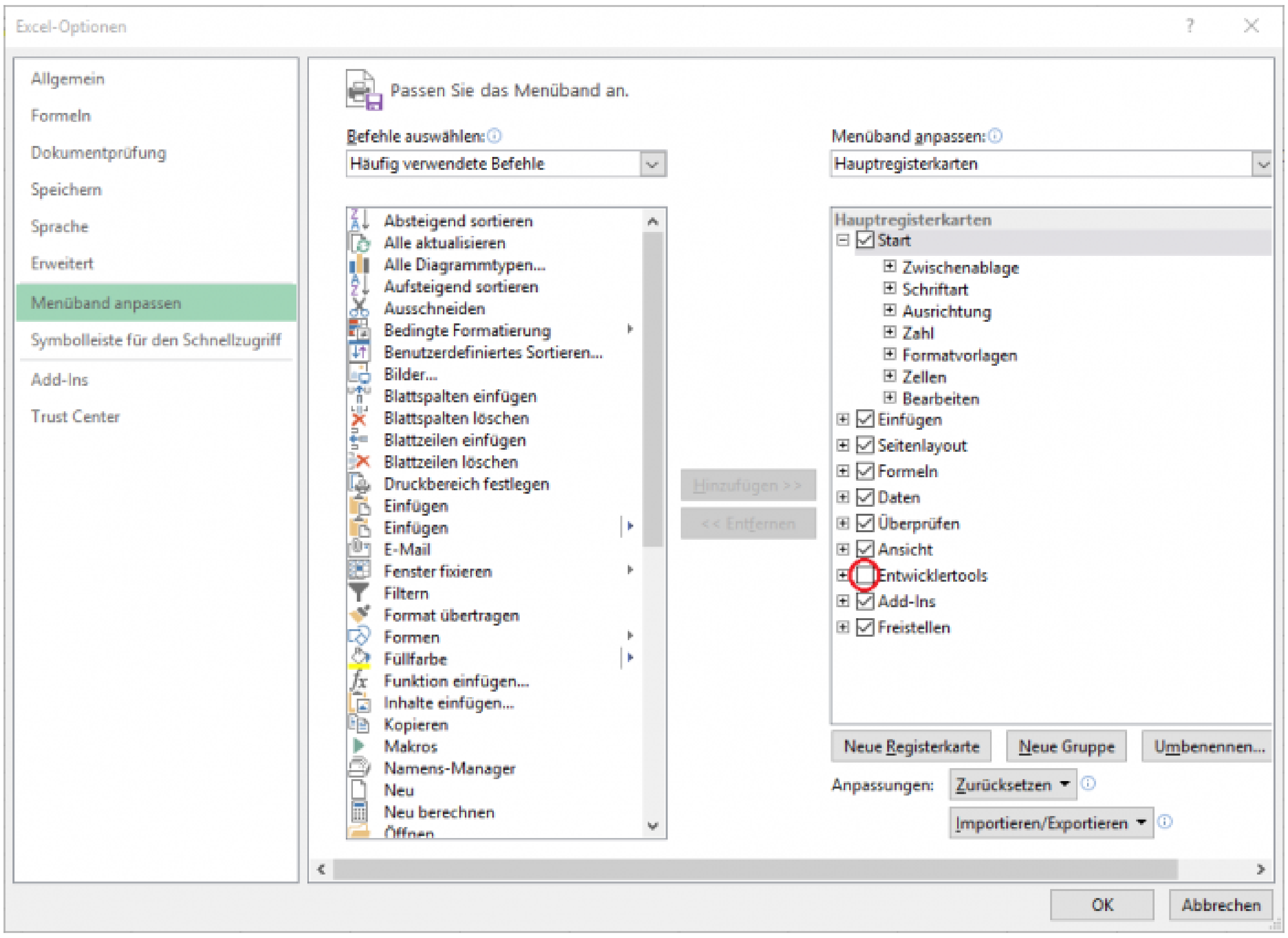
Help:
DB#.DBB # = Data Block#.Data Block Byte #
DB#.DBW # = Data Block#.Data Block Word #
DB#.DBD # = Data Block#.Data Block Doubleword #
# = Address

Getting started with the IP-S7-Link .Net Advanced COM in Microsoft Excel

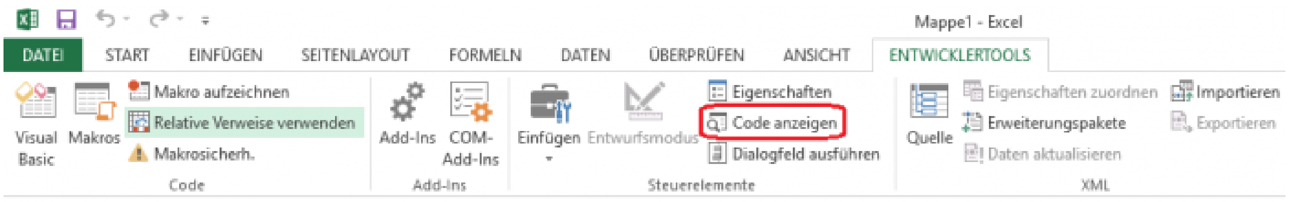
© by TIS

After installing the IP-S7-Link .Net Advanced COM using the Windows Installer Package provided you need to perform the following steps to integrate the library within your Microsoft Excel Workbook to take use of its API in your VBA Macros:

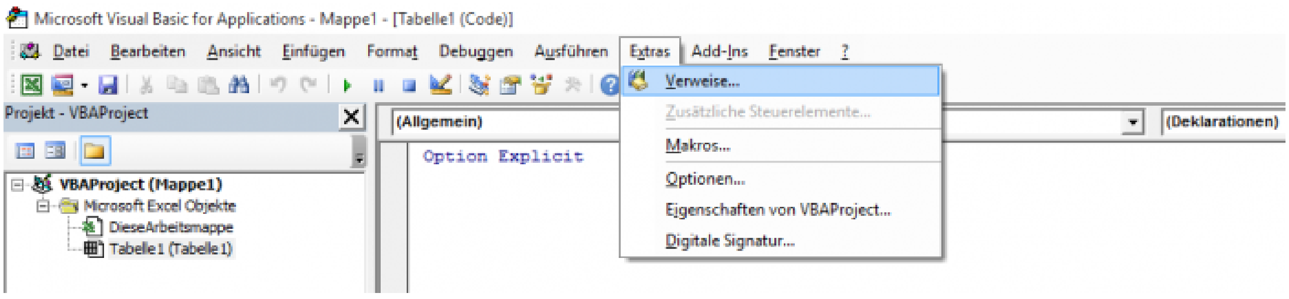
1. Ensure that Excel does display the developer tools menu band (in german: Entwicklertools). To do so go to the Excel Options (in german: Optionen) and select the menu band configuration (in german: Menüband anpassen). Then activate the developer tools menu band.



2. Select the developer tools menu band (in german: Entwicklertools). Then click on view code (in german: Code anzeigen). This does open the integrated development environment (IDE) to develop your Visual Basic for Applications (VBA) Macros.

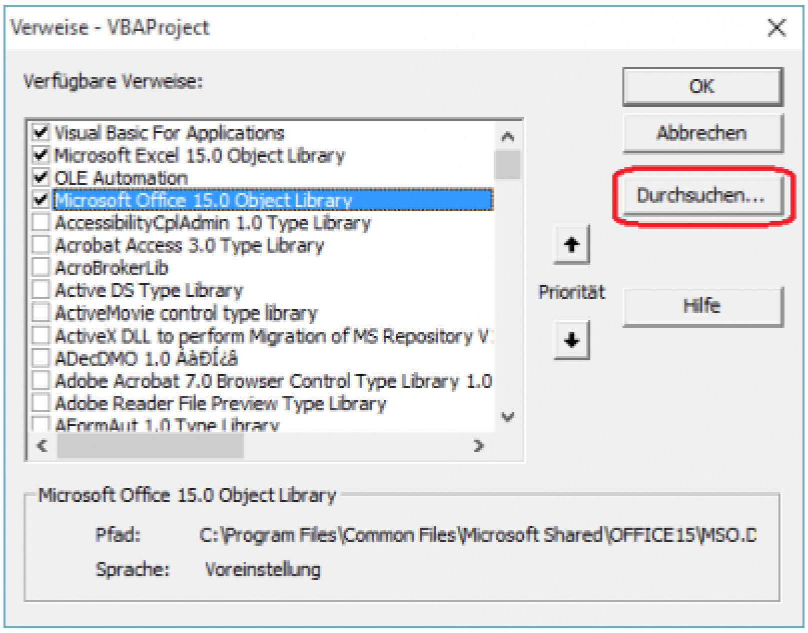


3. Select the Extras menu item (in german: Extras). Then click on the references menu item (in german: Verweise). This does open the references dialog. Using this dialog you are able to add additional library's to your Excel Workbook to take use of their functionality provided developing your VBA Macro.



```
Option Explicit
```

4. Select the browse button (in german: Durchsuchen) to pick up the type library file (*.tlb) which describes the library (*.dll). This file is used by the IDE to recognize the classes, methods, etc. provided by the library.



Verfügbare Verweise:

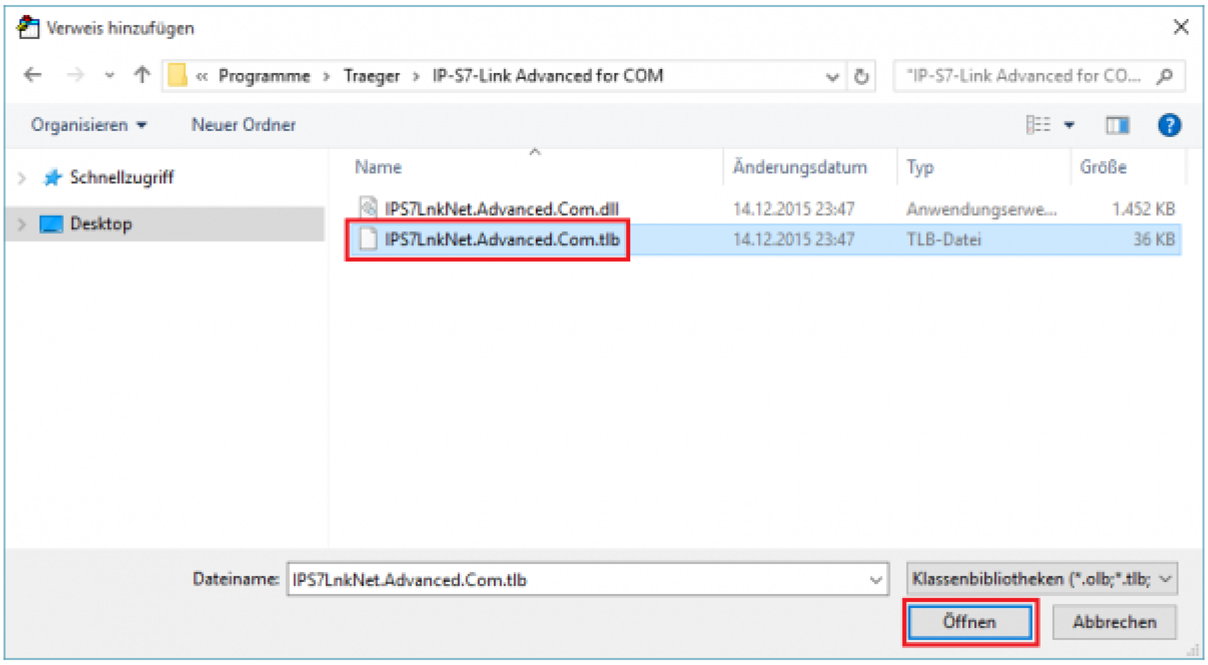
- [✓] Visual Basic For Applications
- [✓] Microsoft Excel 15.0 Object Library
- [✓] OLE Automation
- [✓] Microsoft Office 15.0 Object Library
- [ ] AccessibilityCplAdmin 1.0 Type Library
- [ ] Acrobat Access 3.0 Type Library
- [ ] AcroBrokerLib
- [ ] Active DS Type Library
- [ ] ActiveMovie control type library
- [ ] ActiveX DLL to perform Migration of MS Repository V
- [ ] ADecDMO 1.0 ÀàĐÍ¿â
- [ ] Adobe Acrobat 7.0 Browser Control Type Library 1.0
- [ ] Adobe Reader File Preview Type Library
- [ ] AFormAut 1.0 Type Library

Microsoft Office 15.0 Object Library

Pfad: C:\Program Files\Common Files\Microsoft Shared\OFFICE15\MSO.D

Sprache: Voreinstellung

5. Navigate to the installation folder selected during the setup procedure here we have the default installation location: C:\Program Files\IP-S7-Link Advanced for COM\. When the appropriate installation folder has been opened select the IPS7LnkNet.Advanced.Com.tlb file and confirm with the open button (in german: Öffnen).

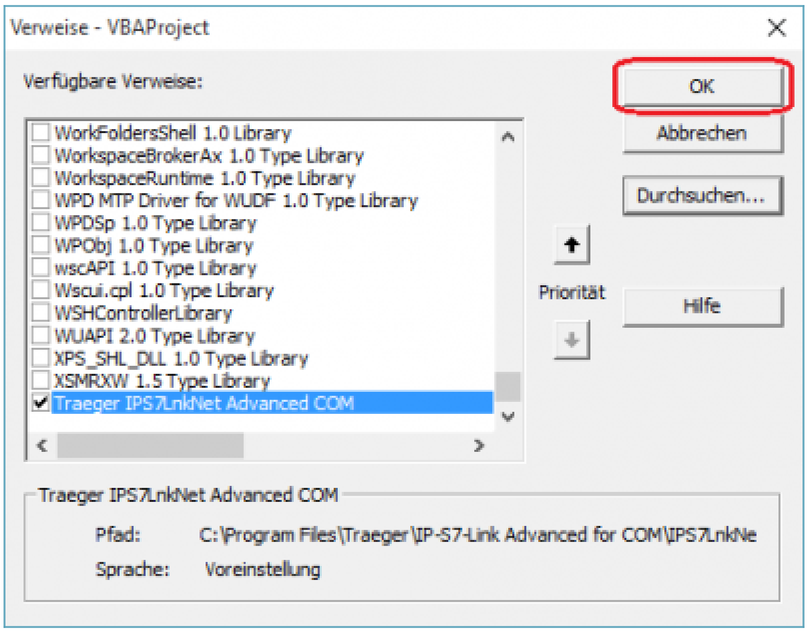


Name	Änderungsdatum	Typ	Größe
IPS7LnkNet.Advanced.Com.dll	14.12.2015 23:47	Anwendungserwe...	1.452 KB
IPS7LnkNet.Advanced.Com.tlb	14.12.2015 23:47	TLB-Datei	36 KB

Dateiname: IPS7LnkNet.Advanced.Com.tlb

Klassenbibliotheken (*.olb;*.tlb;

6. After selecting the IPS7LnkNet.Advanced.Com.tlb you will now find an entry in the references list with the name "IPS7LnkNet Advanced COM" which is already checked. In case it is not just set your tick. Finally confirm and add the new reference to your Excel Workbook by clicking the OK button.



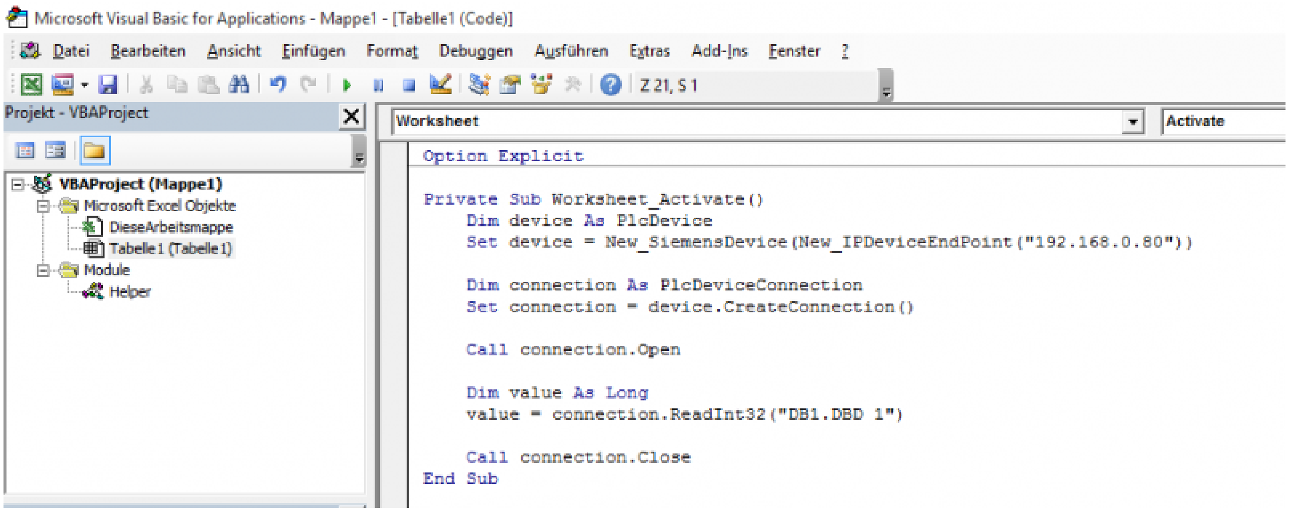
- [ ] WorkFoldersShell 1.0 Library
- [ ] WorkspaceBrokerAx 1.0 Type Library
- [ ] WorkspaceRuntime 1.0 Type Library
- [ ] WPD MTP Driver for WUDF 1.0 Type Library
- [ ] WPDSp 1.0 Type Library
- [ ] WPObj 1.0 Type Library
- [ ] wscAPI 1.0 Type Library
- [ ] Wscui.cpl 1.0 Type Library
- [ ] WSHControllerLibrary
- [ ] WUAPI 2.0 Type Library
- [ ] XPS_SHL_DLL 1.0 Type Library
- [ ] XSMRXW 1.5 Type Library
- [✓] Traeger IPS7LnkNet Advanced COM

Traeger IPS7LnkNet Advanced COM

Pfad: C:\Program Files\Traeger\IP-S7-Link Advanced for COM\IPS7LnkNe

Sprache: Voreinstellung

7. After the required reference has been added you are now able to get started with our samples.


Microsoft Visual Basic for Applications - Mappe1 - [Tabelle1 (Code)]
Datei Bearbeiten Ansicht Einfügen Format Debuggen Ausführen Extras Add-Ins Fenster ?
Z 21, S 1
Projekt - VBAProject
VBAProject (Mappe1)
Microsoft Excel Objekte
DieseArbeitsmappe
Tabelle1 (Tabelle1)
Module
Helper
Worksheet
Activate

```
Option Explicit

Private Sub Worksheet_Activate()
    Dim device As PlcDevice
    Set device = New_SiemensDevice(New_IPDeviceEndPoint("192.168.0.80"))

    Dim connection As PlcDeviceConnection
    Set connection = device.CreateConnection()

    Call connection.Open

    Dim value As Long
    value = connection.ReadInt32("DB1.DBD 1")

    Call connection.Close
End Sub
```

IP-S7-Link .Net Advanced COM Code Snippets

© by TIS

The following code snippets imply that you are using the file Helper.bas contained in the samples included within the installation of the library.

Device Provider

Use one of the different PLC device providers by instantiating an instance of the appropriate PlcDevice class derivates.

```
Dim device As PlcDevice
Set device = New_SiemensDevice(New_IPDeviceEndPoint("192.168.0.80"))
```

Device Configuration

After an instance of (for e.g.) the SiemensDevice class has been created just use the properties provided by the class to setup the appropriate device metadata.

```
Dim device As SiemensDevice
Set device = New_SiemensDevice(New_IPDeviceEndPoint("192.168.0.80"))

device.Type = SiemensDeviceType_S71200
device.ChannelType = SiemensChannelType_OperationPanel
```

In the most scenarios its already enough to create a device using the appropriate device type in the constructor of the device, because the default channel does mostly match the common needs.

```
Dim device As SiemensDevice
Set device = New_SiemensDevice(New_IPDeviceEndPoint("192.168.0.80"),
SiemensDeviceType_S71200)
```

Device End Point

The framework does provide the ability to define different types of end points through the PlcDeviceEndPoint class. The most common end point implementation to use is the IPDeviceEndPoint class. Using this class it is possible to specify the IP address and optionally the rack and slot number of the PLC.

```
Dim endPoint As PlcDeviceEndPoint
Set endPoint = New_IPDeviceEndPoint("192.168.0.80")

' Alternatively also specify the rack.
Dim endPointWithRack As PlcDeviceEndPoint
Set endPointWithRack = New_IPDeviceEndPoint("192.168.0.80", 0)

' Alternatively also specify the rack and slot.
Dim endPointWithRackSlot As PlcDeviceEndPoint
Set endPointWithRackSlot = New_IPDeviceEndPoint("192.168.0.80", 0, 2)
```

Device Connection

After creating an instance of one of the PlcDevice class derivates just create a new connection associated with the device represented. Creating an instance using this factory method will then return the device provider dependent implementation of the PlcDeviceConnection class.

```
Dim device As PlcDevice
Set device = New_SiemensDevice(New_IPDeviceEndPoint("192.168.0.80"))

Dim connection As PlcDeviceConnection
Set connection = device.CreateConnection()
```

Handle Connection State

To retrieve the current connectivity state of the PlcDeviceConnection class instance use the State property to get the according PlcDeviceConnectionState enumeration member.

```
If (connection.State = PlcDeviceConnectionState_Connected) Then
    ' ...
End If
```

To handle the state transitions use one or more of the state specific events of the PlcDeviceConnection class instance.

```
Private WithEvents m_connection As PlcDeviceConnection
```

To handle the event declare the matching event handler as follows:

```
Private Sub m_connection_Connected(ByRef sender As Object, ByVal e As IComEventArgs)
    Debug.Print "Connection established!"
End Sub
```

Read Values

To read a single value use one of the Read methods of the PlcDeviceConnection class.

```
Dim value As Long
value = connection.ReadInt32("DB1.DBD 1")
```

To read an array of values use the additional read overloads to specify the number of items to read as an array as follows:

```
Dim values() As Long
values = connection.ReadInt32Array("DB1.DBD 1", 3)
```

Write Values

To write a single value use one of the Write methods of the PlcDeviceConnection class.

```
Call connection.WriteInt32("DB1.DBD 1", 123)
```

To write an array of values use the additional write overloads to specify an array of items to write as follows:

```
Dim values(3) As Long
values(0) = 123
values(1) = 456
values(2) = 789

Call connection.WriteInt32Array("DB1.DBD 1", values)
```

IP-S7-Link .Net Advanced COM Frequently Asked Questions (FAQs)

Microsoft Excel Exceptions

Situation:
Excel does not longer recognize an imported type or method.

Reason
Through uninstalling and re-installing a Excel imported library this may affect the library registration so Excel may not longer be able to resolve some type and method information contained in the library.

Solution

- Open the references dialog.
- Lookup the imported library its type or method was highlighted by Excel.
- Remove the reference to the library from your Excel workbook.
- Commit the removement using OK.
- Save the Excel workbook changes (this is important).
- Then re-open the references dialog.
- Select the previously removed library.
- Commit this step using OK.
- Save the Excel workbook changes.
- Now everything should work again as expected.

Table of Contents