

Manual | EN

TX1000

TwinCAT 2 | ADS Silverlight/Expression



TwinCAT 2 | Connectivity



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

© Beckhoff Automation GmbH & Co. KG, Germany.

The reproduction, distribution and utilization of this document as well as the communication of its contents to others without express authorization are prohibited.

Offenders will be held liable for the payment of damages. All rights reserved in the event of the grant of a patent, utility model or design.

1.2 For your safety

Safety regulations

Read the following explanations for your safety.

Always observe and follow product-specific safety instructions, which you may find at the appropriate places in this document.

Exclusion of liability

All the components are supplied in particular hardware and software configurations which are 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 technology who are familiar with the applicable national standards.

Signal words

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

Personal injury warnings

DANGER

Hazard with high risk of death or serious injury.

WARNING

Hazard with medium risk of death or serious injury.

CAUTION

There is a low-risk hazard that could result in medium or minor injury.

Warning of damage to property or environment

NOTICE

The environment, equipment, or data may be damaged.

Information on handling the product



This information includes, for example:
recommendations for action, assistance or further information on the product.

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 Overview

Silverlight

Silverlight for Embedded

- target platform: Windows CE 6 R3
- implementation: C++

Silverlight

- target platforms: XP, XPE, WES, Vista, Win 7
- implementation: JavaScript, Visual C#

Samples Silverlight

| Description | Sample |
|---|---|
| Sample 1: Machine Silverlight for Embedded C++ Sample [▶ 7] | https://infosys.beckhoff.com/content/1033/tcsample_expression/Resources/12493869707/.zip |
| Sample 2: Machine Silverlight JavaScript Sample [▶ 22] | https://infosys.beckhoff.com/content/1033/tcsample_expression/Resources/12493871115/.zip |

Windows Presentation Foundation

- target platforms: XP, XPE, WES, Vista, Win 7
- implementation: Visual C#, Visual Basic

Samples WPF

| Description | Sample |
|--|---|
| Sample 1: Machine WPF C# Sample [▶ 29] | https://infosys.beckhoff.com/content/1033/tcsample_expression/Resources/12493866891/.zip |
| Sample 2: Machine WPF Visual Basic Sample [▶ 36] | https://infosys.beckhoff.com/content/1033/tcsample_expression/Resources/12493868299/.zip |

Documents about this

- 📄 [sampleexpressionvista.zip \(Resources/zip/12493866891.zip\)](#)
- 📄 [sampleexpressionvistavb.zip \(Resources/zip/12493868299.zip\)](#)

3 Samples Silverlight

Silverlight

Silverlight for Embedded

- target platform: Windows CE 6 R3
- implementation: C++

Silverlight

- target platforms: XP, XPE, WES, Vista, Win 7
- implementation: JavaScript, Visual C#

Samples Silverlight

| Description | Sample |
|---|---|
| Sample 1: Machine Silverlight for Embedded C++ Sample [▶ 7] | https://infosys.beckhoff.com/content/1033/tcsample_expression/Resources/12493869707/.zip |
| Sample 2: Machine Silverlight JavaScript Sample [▶ 22] | https://infosys.beckhoff.com/content/1033/tcsample_expression/Resources/12493871115/.zip |

3.1 Sample for a machine with Microsoft Silverlight for Windows Embedded

A new item in Windows Embedded CE 6.0 R3 is Silverlight for Windows Embedded. With this new technology, user interfaces of CE devices can now be written in XAML and designed with tools such as Microsoft Expression Blend. On the basis of the machine sample, the creation of a Silverlight for Windows Embedded application with integration of the ADS components is described here .

Target platform

- Windows CE 6 R3

Implementation

- C++

Required software

- Microsoft Visual Studio 2008
- Microsoft Expression Blend 2 SP1
- TwinCAT 2.11
- Beckhoff HMI 600 SDK

Required hardware

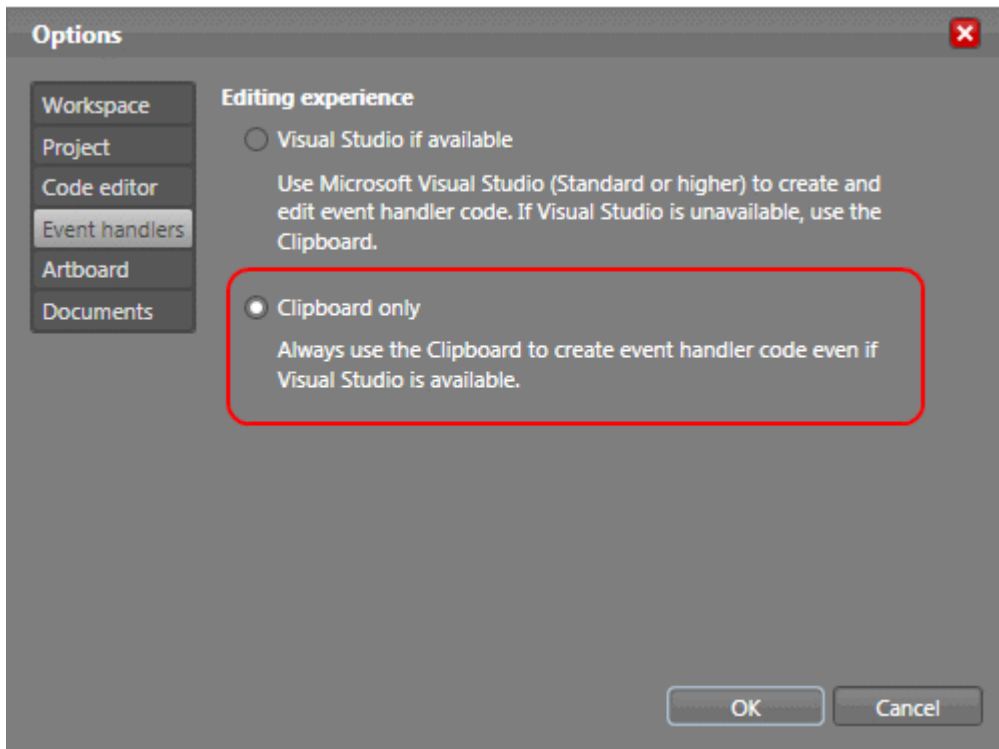
- Windows CE 6.0 R3 device (e.g. CX1020)

First steps...

1. Creating a new Silverlight 2 project:

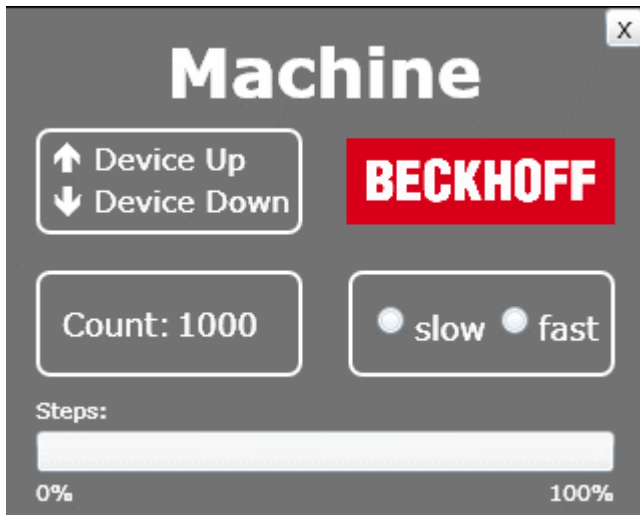
The design of a Silverlight for Windows Embedded application is written in XAML. To this end a Silverlight 2 project is created with Microsoft Expression Blend 2 SP1 via '*File - > New Project*'. The Visual Studio Solution thereby created is not needed in this sample. In addition, the selection of the programming language (Visual C# or Visual Basic) can be ignored. Silverlight for Windows Embedded supports only Visual C++, which is not integrated in Expression Blend. It is therefore also not possible to use the source code

generated by this tool. Disable Visual Studio integration in Expression Blend to avoid the unnecessary automatic generation of Visual C# and Visual Basic code. To do this, select: 'Options->Event handlers' 'Clipboard only'.



2. Creating a user interface

The user interface can now be created in Expression Blend.



In the upper left area, the two outputs can be seen that are also output to the Bus Terminals. The bottom left shows the variable for counting the workpieces. The speed can be set on the right. The 'Steps' display corresponds to the number of cycles. In addition, a button for ending the program is created at the top right.

```
<UserControl xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" Width="319" Height="255">
  <!-- Timelines -->
  <UserControl.Resources>
    <!-- Timeline Device Down -->
    <Storyboard x:Name="timelineDeviceDown">
      <ColorAnimationUsingKeyFrames BeginTime="00:00:00" Storyboard.TargetName="txtDeviceDown"
Storyboard.TargetProperty="(TextBlock.Foreground).(SolidColorBrush.Color)">
        <SplineColorKeyFrame KeyTime="00:00:00.4000000" Value="#FFF000"/>
      </ColorAnimationUsingKeyFrames>
    </Storyboard>
  </UserControl.Resources>
</UserControl>
```



```

</Storyboard>
<!-- Timeline Device Up -->
<Storyboardx:Name="timelineDeviceUp">
  <ColorAnimationUsingKeyFramesBeginTime="00:00:00" Storyboard.TargetName="txtDeviceUp"
Storyboard.TargetProperty="(TextBlock.Foreground).(SolidColorBrush.Color)">
    <SplineColorKeyFrameKeyTime="00:00:00.4000000" Value="#FFFFFF0000"/>
  </ColorAnimationUsingKeyFrames>
</Storyboard>
<!-- Timeline Engine -->
<Storyboardx:Name="timelineEngine"/>
</UserControl.Resources>
<!-- Beginn der Layout Beschreibung -->
<Gridx:Name="LayoutRoot" Background="#FF595959">
  <!-- Title Machine -->
  <TextBlockText="Machine" Margin="80,8.438,80,0" VerticalAlignment="Top" FontWeight="Bold"
Foreground="#FFFFFF" FontSize="34"/>
  <!-- Device Up / Device Down -->
  <GridMargin="15,60,0,0" HorizontalAlignment="Left" VerticalAlignment="Top" Height="53"
Width="132.532">
    <RectangleFill="{x:Null}" Stroke="#FFFFFF" StrokeThickness="2" RadiusX="6"
RadiusY="6"/>
    <TextBlockText="Device Up" Margin="28.823,5.396,-8.823,0" VerticalAlignment="Top"
Foreground="#FFFFFF" FontSize="16"/>
    <TextBlockText="Device Down" Margin="29.002,0,-9.002,4.994" VerticalAlignment="Bottom"
Foreground="#FFFFFF" FontSize="16"/>
    <TextBlockx:Name="txtDeviceDown" Text="è" Margin="7.517,0,0,4.998"
HorizontalAlignment="Left" VerticalAlignment="Bottom" FontFamily="Wingdings" FontWeight="Bold"
Foreground="#FFFFFF" FontSize="16"/>
    <TextBlockx:Name="txtDeviceUp" Text="é" Margin="7.517,5.497,0,0"
HorizontalAlignment="Left" VerticalAlignment="Top" FontFamily="Wingdings" FontWeight="Bold"
Foreground="#FFFFFF" FontSize="16"/>
  </Grid>
  <!-- Counter -->
  <GridMargin="15,0,0,71" HorizontalAlignment="Left" VerticalAlignment="Bottom" Height="53"
Width="133">
    <RectangleFill="{x:Null}" Stroke="#FFFFFF" StrokeThickness="2" RadiusX="6"
RadiusY="6"/>
    <StackPanelMargin="12.991,16,0,16" HorizontalAlignment="Left" Orientation="Horizontal"
Width="113">
      <TextBlockText="Count:" Width="54.824" Foreground="#FFFFFF" FontSize="16"/>
      <TextBlockx:Name="txtCount" Margin="2,0,0,0" Foreground="#FFFFFF" FontSize="16"/>
    </StackPanel>
  </Grid>
  <!-- Speed -->
  <GridMargin="0,0,15,71" Height="53" HorizontalAlignment="Right" VerticalAlignment="Bottom"
Width="132.532">
    <RectangleFill="{x:Null}" Stroke="#FFFFFF" StrokeThickness="2" RadiusX="6"
RadiusY="6"/>
    <StackPanelMargin="12.988,0,3.012,0" Orientation="Horizontal">
      <RadioButtonx:Name="radSpeedSlow" Content="slow" Margin="0,0,6,0"
Foreground="#FFFFFF" FontSize="16" Height="19.496"/>
      <RadioButtonx:Name="radSpeedFast" Content="fast" Foreground="#FFFFFF"
FontSize="16" Height="19.496"/>
    </StackPanel>
  </Grid>
  <!-- Steps -->
  <GridMargin="15,0,15,5" VerticalAlignment="Bottom" Height="57">
    <ProgressBarx:Name="prgSteps" Margin="0,18,0,18" Maximum="25"/>
    <TextBlockText="Steps:" HorizontalAlignment="Left" VerticalAlignment="Top"
Foreground="#FFFFFF"/>
    <TextBlockText="0%" HorizontalAlignment="Left" VerticalAlignment="Bottom"
Foreground="#FFFFFF"/>
    <TextBlockText="100%" HorizontalAlignment="Right" VerticalAlignment="Bottom"
Foreground="#FFFFFF"/>
  </Grid>
  <!-- Close Button -->
  <Buttonx:Name="butClose" Content="X" HorizontalAlignment="Right" VerticalAlignment="Top"
Height="20" Width="20"/>
  <!-- Beckhoff Logo -->
  <ImageSource="beckhoff_logo_white.jpg" Margin="0,64.502,15,0" HorizontalAlignment="Right"
VerticalAlignment="Top" Height="43.151" Width="133.745" Stretch="Fill"/>
</Grid>
</UserControl>

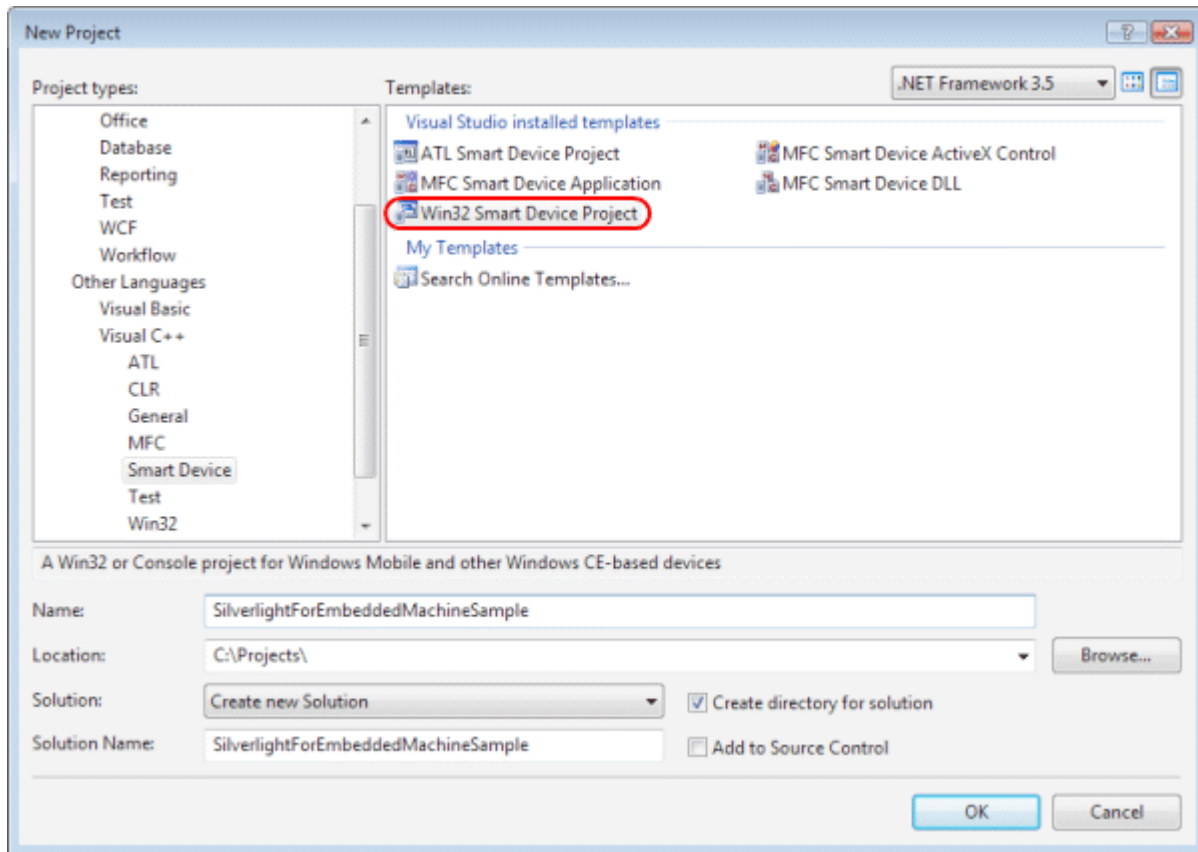
```

3. Create new Win32 Smart Device project

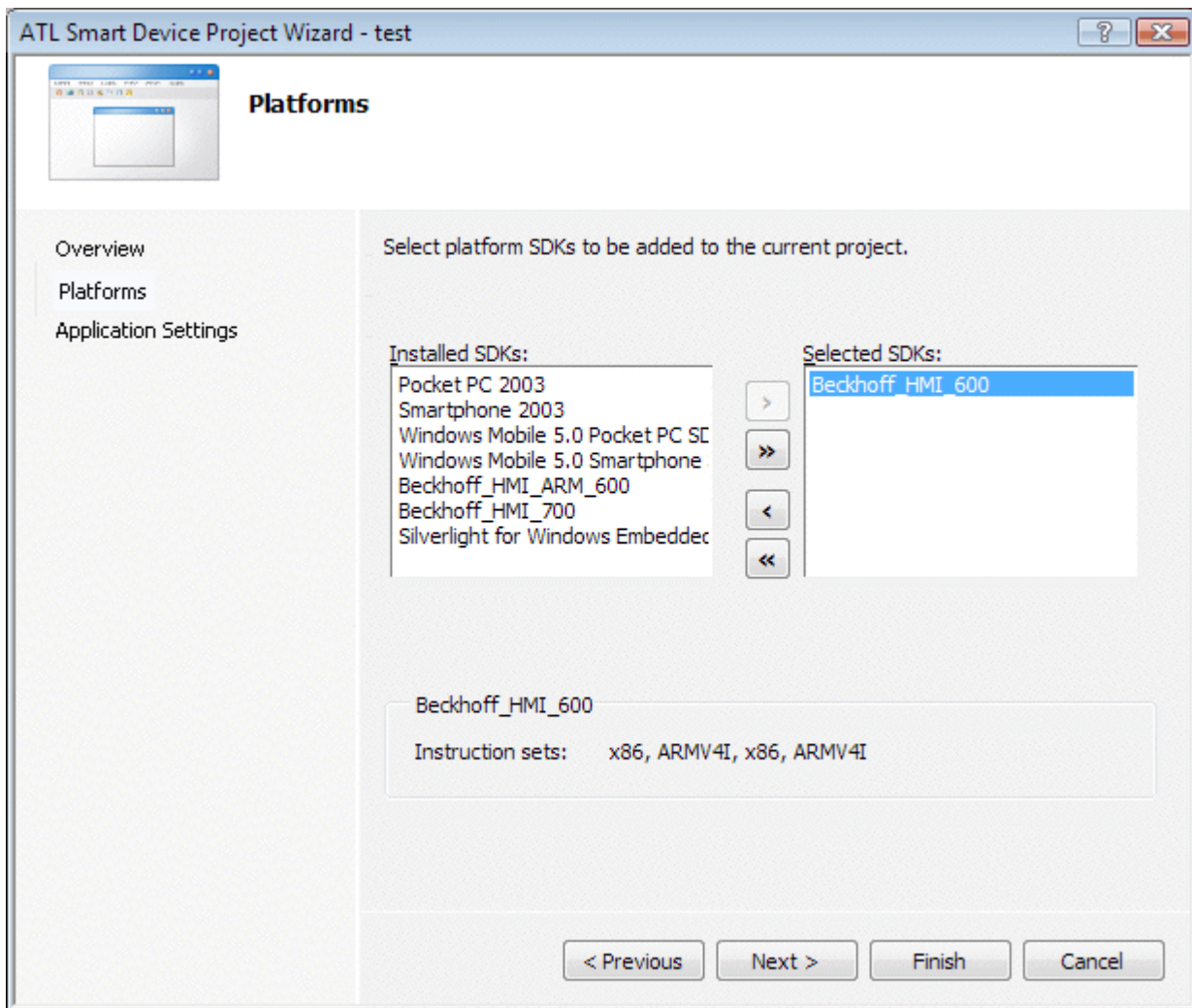
A new Win32 Smart Device project must now be created in Visual Studio 2008.



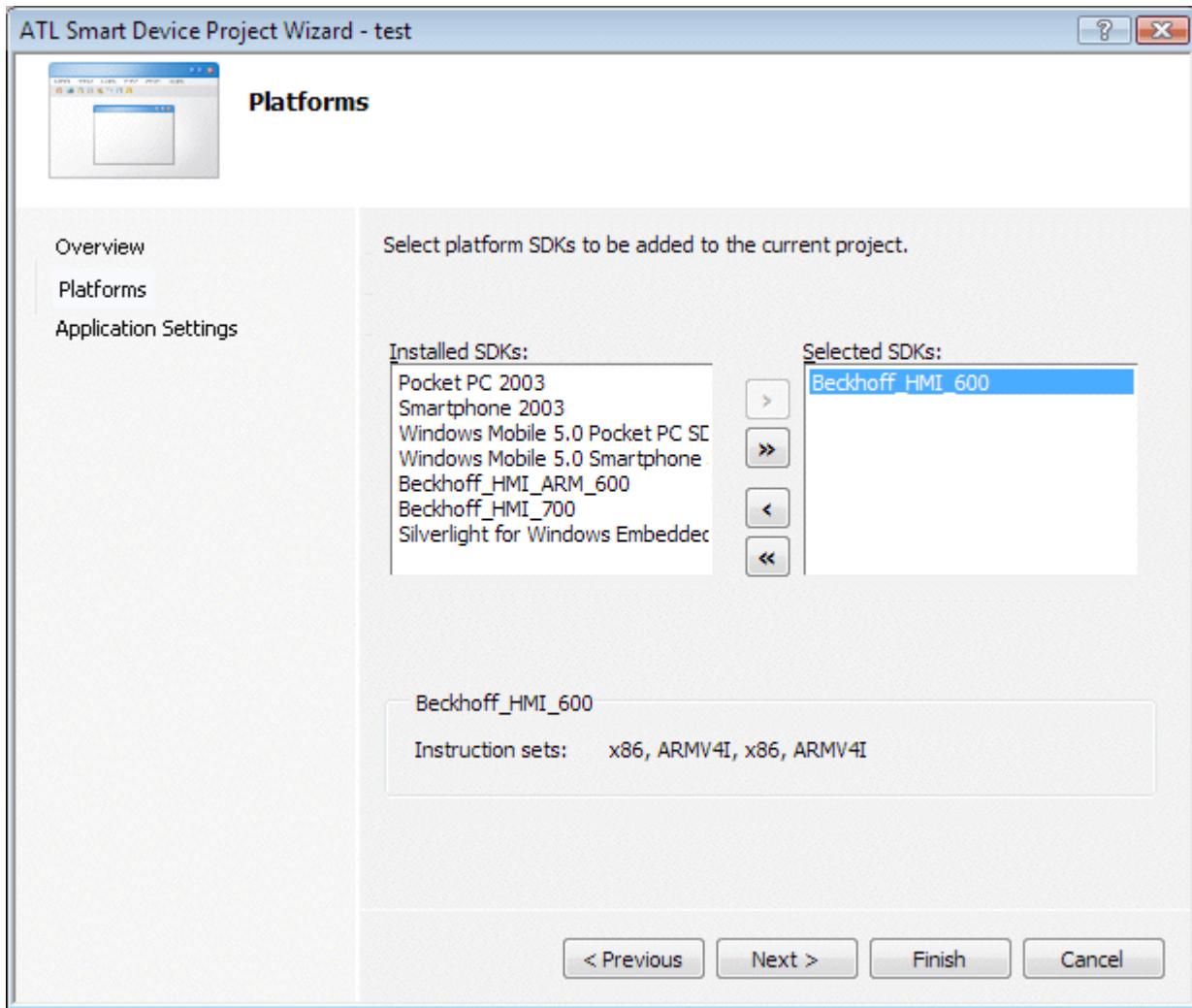
If the Beckhoff HMI 600 SDK is not yet installed on the computer, install it before creating a new Visual Studio project.



The Platform SDK of this project is the Beckhoff HMI 600 SDK.

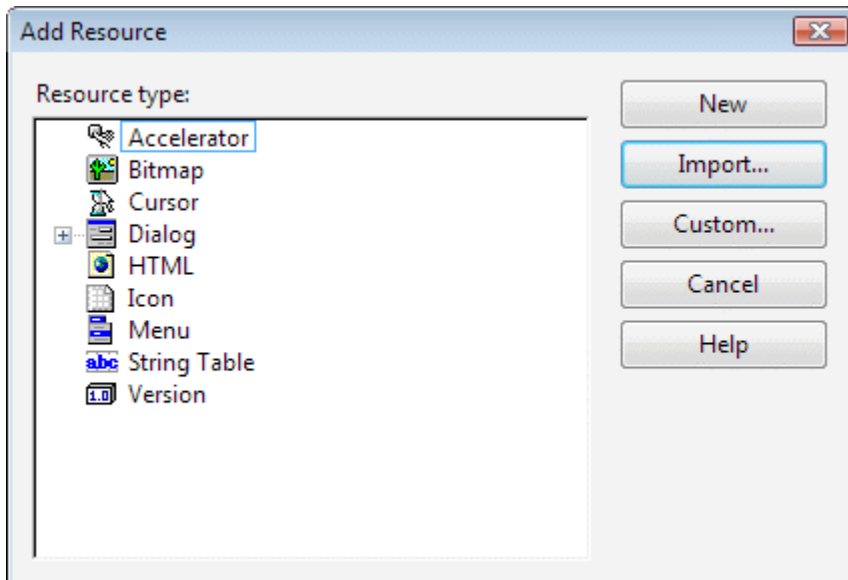


Executable (EXE) is selected as the server type.

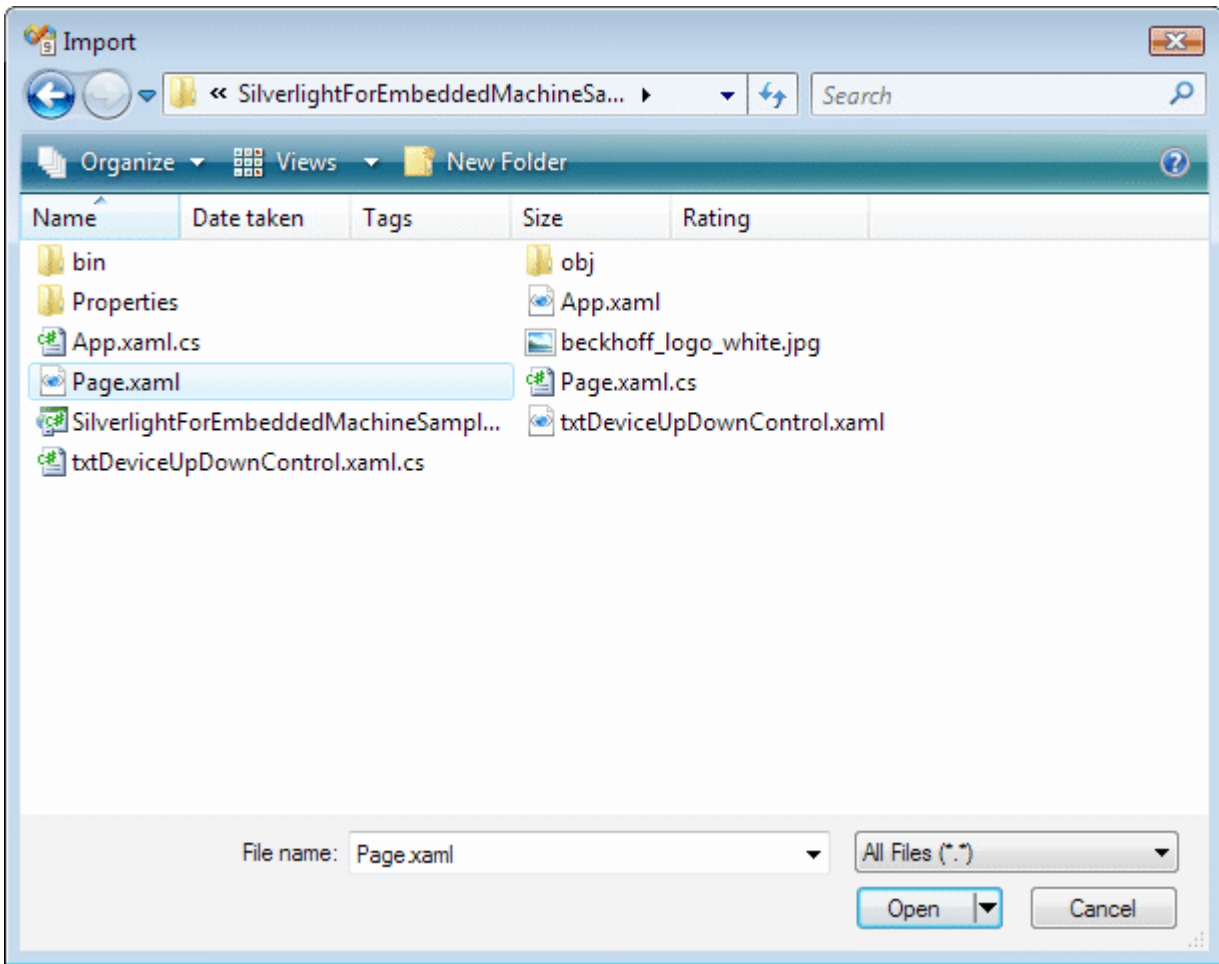


4. Integrating the XAML file as a resource

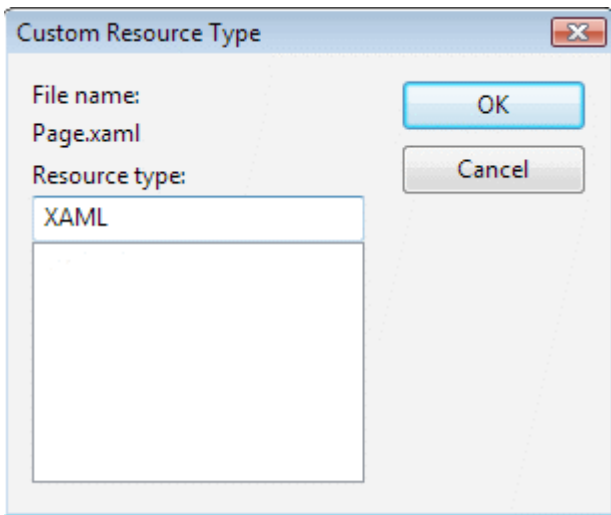
The user interface designed with Expression Blend can be integrated in the new project. To do this, open the resource file (.rc). A right mouse click on the resource in the Resource View tab and the selection of 'Add -> Resource...' opens a dialog box via which the XAML file can be integrated.



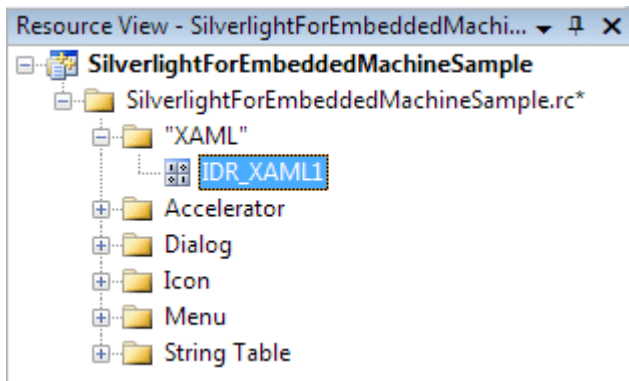
The XAML file can be imported into the project with the aid of the dialog box.



Specify XAML as the resource type.



The standard resource ID (IDR_XAML1) can be retained in this sample. In your own projects, however, it makes sense to rename them.



5. Creating an AdsHelper Class

Most of the Ads communication can be outsourced to a separate class, here called AdsHelper.

AdsHelper.h

The TcAds headers are included in the AdsHelper header. Care must be taken to specify the correct path. In addition, the headers depend on the processor type. In the sample code the X86 headers are attached.

```
#include "..\AdditionalFiles\TcpAdsApiCe\include\TcAdsDef.h" #include "..\AdditionalFiles\TcpAdsApiCe\include\TcAdsAPI.h"
```

Furthermore, the declarations are made in the header.

```
typedef enum E_NOTIFICATION_IDENT
{
    engine = 0,
    deviceUp = 1,
    deviceDown = 2,
    steps,
    count,
    switchSpeed
};

long AdsGetVarHandle(AmsAddr* pServerAddr, const char* szVarname, long* pHandle);
long AdsFreeVarHandle(AmsAddr* pServerAddr, long handle);
long AdsGerVarHandleEx(long port, AmsAddr* pServerAddr, const char* szVarname, long* pHandle);
long AdsFreeVarHandleEx(long port, AmsAddr* pServerAddr, long handle);

long SpeedSlowEx(long port, AmsAddr* pServerAddr);
long SpeedFastEx(long port, AmsAddr* pServerAddr);

long connect(long port, AmsAddr* pServerAddr);
long disconnect(long port, AmsAddr* pServerAddr);
```

AdsHelper.cpp

The headers StdAfx.h and AdsHelper.h must be integrated in the AdsHelper.cpp,

```
#include "StdAfx.h" #include "AdsHelper.h"
```

and afterwards the global variables defined.

```
long hEngine, hDeviceUp, hDeviceDown, hSteps, hCount, hSwitch;
unsigned long hEngineNotification, hDeviceUpNotification, hDeviceDownNotification,
    hStepsNotification, hCountNotification, hSwitchNotification;
```

The methods AdsGetVarHandle and AdsGetVarHandleEx serve to create handles for PLC variables

```
long AdsGetVarHandle(AmsAddr* pServerAddr, const char* szVarname, long* pHandle)
{
    if (pHandle == NULL || pServerAddr == NULL)
        return E_POINTER;

    unsigned long read = 0;
    long nErr =
        AdsSyncReadWriteReqEx(pServerAddr, ADSIGRP_SYM_HNDBYNAME, 0x0,
            sizeof(long), pHandle, strlen(szVarname), (char*)szVarname, &read);
```

```

    return nErr;
}

long AdsGetVarHandleEx(long port, AmsAddr* pServerAddr, const char* szVarname, long* pHandle)
{
    if(pHandle == NULL || pServerAddr == NULL)
        return E_POINTER;

    unsigned long read = 0;

    long nErr =
        AdsSyncReadWriteReqEx2(port, pServerAddr, ADSIGRP_SYM_HNDBYNAME, 0x0,
            sizeof(long), pHandle, strlen(szVarname), (char*)szVarname, &read);

    return nErr;
}

```

Handles for PLC variables are released again with `AdsFreeVarHandle` and `AdsFreeVarHandleEx`.

```

long AdsFreeVarHandle(AmsAddr* pServerAddr, long handle)
{
    return AdsSyncWriteReq(pServerAddr, ADSIGRP_SYM_RELEASEHND, 0,
        sizeof(handle), &handle);
}

long AdsFreeVarHandleEx(long port, AmsAddr* pServerAddr, long* pHandle)
{
    return AdsSyncWriteReqEx(port, pServerAddr, ADSIGRP_SYM_RELEASEHND, 0,
        sizeof(pHandle), &pHandle);
}

```

The following two methods write the PLC variable '.switch' and in doing so set the speed to slow or fast.

```

long SpeedSlowEx(long port, AmsAddr* pServerAddr)
{
    // Handle der SPS-Variable ".switch" erstellen.
    long handleSpeedSlow = 0;
    long adserror = AdsGetVarHandleEx(port, pServerAddr, ".switch", &handleSpeedSlow);

    // Die SPS-Variable ".switch" auf FALSE setzen.
    bool datafalse = false;
    adserror = AdsSyncWriteReqEx(port, pServerAddr, ADSIGRP_SYM_VALBYHND,
        handleSpeedSlow, 0x1, &datafalse);

    // Handle der SPS-Variable ".switch" freigeben
    adserror = AdsFreeVarHandleEx(port, pServerAddr, handleSpeedSlow);
    return adserror;
}

long SpeedFastEx(long port, AmsAddr* pServerAddr)
{
    // Handle der SPS-Variable ".switch" erstellen.
    long handleSpeedFast = 0;
    long adserror = AdsGetVarHandleEx(port, pServerAddr, ".switch", &handleSpeedFast);

    // Die SPS-Variable ".switch" auf TRUE setzen.
    bool datatrue = true;
    adserror = AdsSyncWriteReqEx(port, pServerAddr, ADSIGRP_SYM_VALBYHND,
        handleSpeedFast, 0x1, &datatrue);

    // Handle der SPS-Variable ".switch" freigeben
    adserror = AdsFreeVarHandleEx(port, pServerAddr, handleSpeedFast);
    return adserror;
}

```

In the connect method, a connection to the variables is created in the PLC.

```

long connect (long port, AmsAddr* addr, PAdsNotificationFuncEx Callback)
{
    // Attribute der Notification festlegen
    AdsNotificationAttrib attr;
    attr.cbLength = 2;
    attr.nTransMode = ADSTRANS_SERVERCYCLE;
    attr.nMaxDelay = 100000000; // = 1 sec
    attr.nCycleTime = 100000; // = 0,5 sec // Handles der SPS-
    Variablen holen
    long adserr = AdsGetVarHandleEx(port, addr, ".engine", &hEngine);

    if(adserr == 0)

```

```

adserr = AdsGetVarHandleEx(port, addr, ".deviceUp", &hDeviceUp);

if(adserr == 0)
adserr = AdsGetVarHandleEx(port, addr, ".deviceDown", &hDeviceDown);

if(adserr == 0)
adserr = AdsGetVarHandleEx(port, addr, ".steps", &hSteps);

if(adserr == 0)
adserr = AdsGetVarHandleEx(port, addr, ".count", &hCount);

if(adserr == 0)
adserr = AdsGetVarHandleEx(port, addr, ".switch", &hSwitch);

// Überwachung der SPS-Variablen initialisierenif(adserr == 0)
{
attr.cbLength = 1;
adserr = AdsSyncAddDeviceNotificationReqEx(port, addr, ADSIGRP_SYM_VALBYHND, hEngine,
&attr, Callback, engine, &hEngineNotification);
}
if(adserr == 0)
{
attr.cbLength = 1;
adserr = AdsSyncAddDeviceNotificationReqEx(port, addr, ADSIGRP_SYM_VALBYHND, hDeviceUp,
&attr, Callback, deviceUp, &hDeviceUpNotification);
}
if(adserr == 0)
{
attr.cbLength = 1;
adserr = AdsSyncAddDeviceNotificationReqEx(port, addr, ADSIGRP_SYM_VALBYHND, hDeviceDown,
&attr, Callback, deviceDown, &hDeviceDownNotification);
}
if(adserr == 0)
{
attr.cbLength = 1
adserr = AdsSyncAddDeviceNotificationReqEx(port, addr, ADSIGRP_SYM_VALBYHND, hSteps,
&attr, Callback, steps, &hStepsNotification);
}
if(adserr == 0)
{
attr.cbLength = 2;
adserr = AdsSyncAddDeviceNotificationReqEx(port, addr, ADSIGRP_SYM_VALBYHND, hCount,
&attr, Callback, count, &hCountNotification);
}
if(adserr == 0)
{
attr.cbLength = 1;
adserr = AdsSyncAddDeviceNotificationReqEx(port, addr, ADSIGRP_SYM_VALBYHND, hSwitch,
&attr, Callback, switchSpeed, &hSwitchNotification);
}

return adserr;
}

```

When disconnecting the Ads connection, the handles of the PLC variables must be released and the port closed.

```

long disconnect(long port, AmsAddr* addr)
{
// Handles der SPS-Variablen freigeben
AdsFreeVarHandleEx(port, addr, hEngine);
AdsFreeVarHandleEx(port, addr, hDeviceUp);
AdsFreeVarHandleEx(port, addr, hDeviceDown);
AdsFreeVarHandleEx(port, addr, hSteps);
AdsFreeVarHandleEx(port, addr, hCount);
AdsFreeVarHandleEx(port, addr, hSwitch);

// Notifications löschen
AdsSyncDelDeviceNotificationReqEx(port, addr, hEngineNotification);
AdsSyncDelDeviceNotificationReqEx(port, addr, hDeviceUpNotification);
AdsSyncDelDeviceNotificationReqEx(port, addr, hDeviceDownNotification);
AdsSyncDelDeviceNotificationReqEx(port, addr, hStepsNotification);
AdsSyncDelDeviceNotificationReqEx(port, addr, hCountNotification);
AdsSyncDelDeviceNotificationReqEx(port, addr, hSwitchNotification);

// Kommunikationsport schließen

```



```

    AdsPortCloseEx(port);

    return 0;
}

```

6. Processing the source code

The headers are integrated first in the SilverlightForEmbeddedMachineSample.cpp file.

```

#include"pwinuser.h"#include"xamlruntime.h"#include"xrdelegate.h"#include"xrptr.h"    #include"resource.h"

```

The declaration of the variables follows.

```

IXRDelegate<XRMouseButtonEventArgs>* clickdelegate;

UINT exitcode;

IXRVisualHostPtr    vhost;
IXRButtonBasePtr    butClose;
IXRRadioButtonPtr    radSpeedSlow;
IXRRadioButtonPtr    radSpeedFast;
IXRTextBlockPtr     txtDeviceDown;
IXRTextBlockPtr     txtDeviceUp;
IXRTextBlockPtr     txtCount;
IXRProgressBarPtr   prgSteps;

IXRStoryboardPtr    timelineDeviceDown;
IXRStoryboardPtr    timelineDeviceUp;
IXRStoryboardPtr    timelineEngine;

long port;
AmsAddr addr;

unsigned long TimerID;
DWORD EventID;
CRITICAL_SECTION cs;
long event_cnt;
long event_cntold;

void __stdcall Callback(AmsAddr* addr, AdsNotificationHeader* handler, unsigned long User);

```

The timer callback function checks the Ads connection and restores the connection if necessary.

```

VOID CALLBACK MyTimerProc(
    HWND hwnd,        // handle to window for timer messages
    UINT message,     // WM_TIMER message
    UINT idTimer,     // timer identifier
    DWORD dwTimer)    // current system time
{
    if (event_cnt = event_cntold)
    {
        // Handels werden freigegeben und der Port geschlossen
        disconnect(port, &addr);

        // Kommunikationsport auf dem ADS Router öffnen
        port = AdsPortOpenEx();

        addr.port = 0x321;
        long adserror = -1;

        // Neue Verbindung zur SPS herstellen.while(adserror != 0)
        {
            adserror = connect(port, &addr, Callback);
            Sleep(1000);
        }
    }
}

```

```

    event_cntold = event_cnt;
}

```

The following Ads Event handler is called if a PLC variable to which a link exists changes.

```

// ADS-State Callback-
Functionvoid __stdcall Callback(AmsAddr* addr, AdsNotificationHeader* handler, unsigned long User)
{
    event_cnt++;
}

```

The corresponding arrows are colored red or not, depending on whether deviceUp, deviceDown or engine is set to TRUE.

This effect can be improved still further by the use of timelines.

```

if (User == deviceUp)
{
    if (*(bool*)handler->data == true)
    {
        timelineDeviceDown->Stop();
        timelineEngine->Stop();
        timelineDeviceUp->Begin();
    }
}
else if (User == deviceDown)
{
    if (*(bool*)handler->data == true)
    {
        timelineDeviceUp->Stop();
        timelineEngine->Stop();
        timelineDeviceDown->Begin();
    }
}
else if (User == engine)
{
    if (*(bool*)handler->data == true)
    {
        timelineDeviceDown->Stop();
        timelineDeviceUp->Stop();
        timelineEngine->Begin();
    }
}
}

```

steps indicates the number of cycles. The value is output via the progress bar *prgSteps*. For this the data must first be converted to a byte, since the associated PLC variable is of the type byte. Since the progress bar can only transfer data of the type float, conversion to the type float subsequently takes place.

```

else if (User == steps)
{
    prgSteps->SetValue((float)*((byte*)handler->data));
}

```

In the case of *count*, as with *steps*, the data must first be converted to their original data type before they can be converted into a text and transferred to the text block *txtCount*.

```

else if (User == count)
{
    WCHAR text[6];
    wprintf(text, L"%d", *(unsigned short*)handler->data);
    txtCount->SetText(text);
}

```

The output of the speed type is done via RadioButtons. The appropriate radio button is marked depending on the speed.

```

else if (User == switchSpeed)
{
    if (*(bool*)handler->data == true)
    {
        radSpeedFast->SetIsChecked(XRThreeState_Checked);
    }
    else
    {
        radSpeedSlow->SetIsChecked(XRThreeState_Checked);
    }
}
}
}

```

The OnClick event is triggered by the various instances and the name can be used to distinguish which instance was the trigger.

```

class BtnEventHandler
{
public:

    HRESULT OnClick(IXRDependencyObject* source, XRMouseButtonEventArgs* args)
    {
        BSTR name;
        HRESULT hr = NULL;
        source->GetName(&name);

        short state = 0;

        long adserror = 0;

        if (wcscmp(name, L"butClose") == 0)
        {
            // Machine Dialog schließen
            vhost->EndDialog(exitcode);
        }
        if (wcscmp(name, L"radSpeedSlow") == 0)
        {
            // Aufruf der Methode SpeedSlowEx um die Geschwindigkeit auf langsam zu setzen.
            adserror = SpeedSlowEx(port, &addr);
        }
        if (wcscmp(name, L"radSpeedFast") == 0)
        {
            // Aufruf der Methode SpeedFastEx um die Geschwindigkeit auf schnell zu setzen.
            adserror = SpeedFastEx(port, &addr);
        }

        if (adserror != NULL)
        {
            // Die Handels werden freigegeben und der Port geschlossen
            disconnect(port, &addr);

            // Der Kommunikationsport auf dem ADS-Router wird geöffnet
            port = AdsPortOpenEx();

            addr.port = 0x321;

            // Neu Verbindung zur SPS herstellen.
            adserror = connect(port, &addr, Callback);

            Sleep(1000);
        }

        SysFreeString(name);
        return S_OK;
    }
};

```

In the WinMain method, the XMAL runtime must be initialized first. If XamlRuntimeInitialize is successful, then Silverlight for Windows Embedded Runtime is started in the application.

```
int WINAPI WinMain(HINSTANCE hInstance,
                  HINSTANCE hPrevInstance,
                  LPTSTR lpCmdLine,
                  int nCmdShow)
{
    // Initialisierung der XAML Runtimeif (!XamlRuntimeInitialize())
    return -1;
}
```

Each Silverlight for Windows Embedded application has a single "application" object that can be used to access global properties.

To access this object the GetXRApplicationInstance API is used.

```
HRESULT retcode;

// Load an dinit XAML resource
IXRApplicationPtr app;

if (FAILED (retcode=GetXRApplicationInstance(&app)))
    return -1;

if (FAILED (retcode=app->AddResourceModule(hInstance)))
    return -1;
```

After the initialization of the application object, the main window can be created and the administration of the object can be handed over to Silverlight for Windows Embedded.

```
XRWindowCreateParams wp;

ZeroMemory(&wp, sizeof(XRWindowCreateParams));

// Set window styles
wp.Style = WS_BORDER;
wp.pTitle = L"Silverlight for Windows Embedded Machine Sample";
wp.Left = 0;
wp.Top = 0;
wp.AllowsMultipleThreadAccess = true;

XRXamlSource xamlsrc;

xamlsrc.SetResource(hInstance, TEXT("XAML"), MAKEINTRESOURCE(IDR_XAML1));

if (FAILED(retcode=app->CreateHostFromXaml(&xamlsrc, &wp, &vhost)))
    return -1;
```

The object within a Silverlight for Windows Embedded application is organized in object trees. In order to access this object, a pointer to the root element is required.

```
IXRFrameworkElementPtr root;

if (FAILED (retcode=app->CreateHostFromXaml(&xamlsrc, &wp, &vhost)))
    return -1;
```

Creating instances of the controls and timelines.

```
// Get controls by nameif (FAILED(retcode=root->FindName(TEXT("butClose"), &butClose))
    return -1;

if (FAILED(retcode=root->FindName(TEXT("radSpeedSlow", &radSpeedSlow)))
    return -1;

if (FAILED(retcode=root->FindName(TEXT("radSpeedFast", &radSpeedFast)))
    return -1;

if (FAILED(retcode=root->FindName(TEXT("txtDeviceDown", &txtDeviceDown)))
    return -1;

if (FAILED(retcode=root->FindName(TEXT("txtDeviceUp", &txtDeviceUp)))
    return -1;
```

```

if (FAILED(retcode=root->FindName(TEXT("txtCount", &txtCount)))
return -1;

if (FAILED(retcode=root->FindName(TEXT("prgSteps", &prgSteps)))
return -1;

// Get timelines by nameif (FAILED (retcode=root-
>FindName(TEXT("timelineDeviceDown"), &timelineDeviceDown))
return -1;

if (FAILED (retcode=root->FindName(TEXT("timelineDeviceUp"), &timelineDeviceUp)))
return -1;

if (FAILED (retcode=root->FindName(TEXT("timelineEngine"), &timelineEngine)))
return -1;

```

Creating the "RadioButtonGroup" and assigning the two radio buttons from this group.

```

WCHAR groupName[17];
wsprintf(groupName, L"RadioButtonGroup");
radSpeedFast->SetGroupName(groupName);
radSpeedSlow->SetGroupName(groupName);

```

A redirecting object is needed to link the EventHandler with the buttons.

```

BtnEventHandler handler;

// Set the event handler for the buttonsif (FAILED(retcode=CreateDelegate(&handler, &BtnEventHan
dler::OnClick, &clickdelegate))
return -1;

if (FAILED(retcode=butClose->btnAddClickEventHandler(clickdelegate)))
return -1;

if (FAILED(retcode=radSpeedSlow->AddClickEventHandler(clickdelegate)))
return -1;

if (FAILED(retcode=radSpeedFast->AddClickEventHandler(clickdelegate)))
return -1;

```

Integrating the Ads components.

```

long adserror = -1;
port = AdsPortOpenEx();

AdsGetLocalAddressEx(port, &addr);

// connect to the PLC and register callbacks
addr.port = 0x321;
adserror = connect(port, &addr, Callback);

event_cnt = 0;
event_cntold = -1;

// init timer for reconnect
SetTimer(NULL, NULL, 5000, MyTimerProc);

if (FAILED(retcode=vhost->StartDialoge(&exitcode))
return -1;

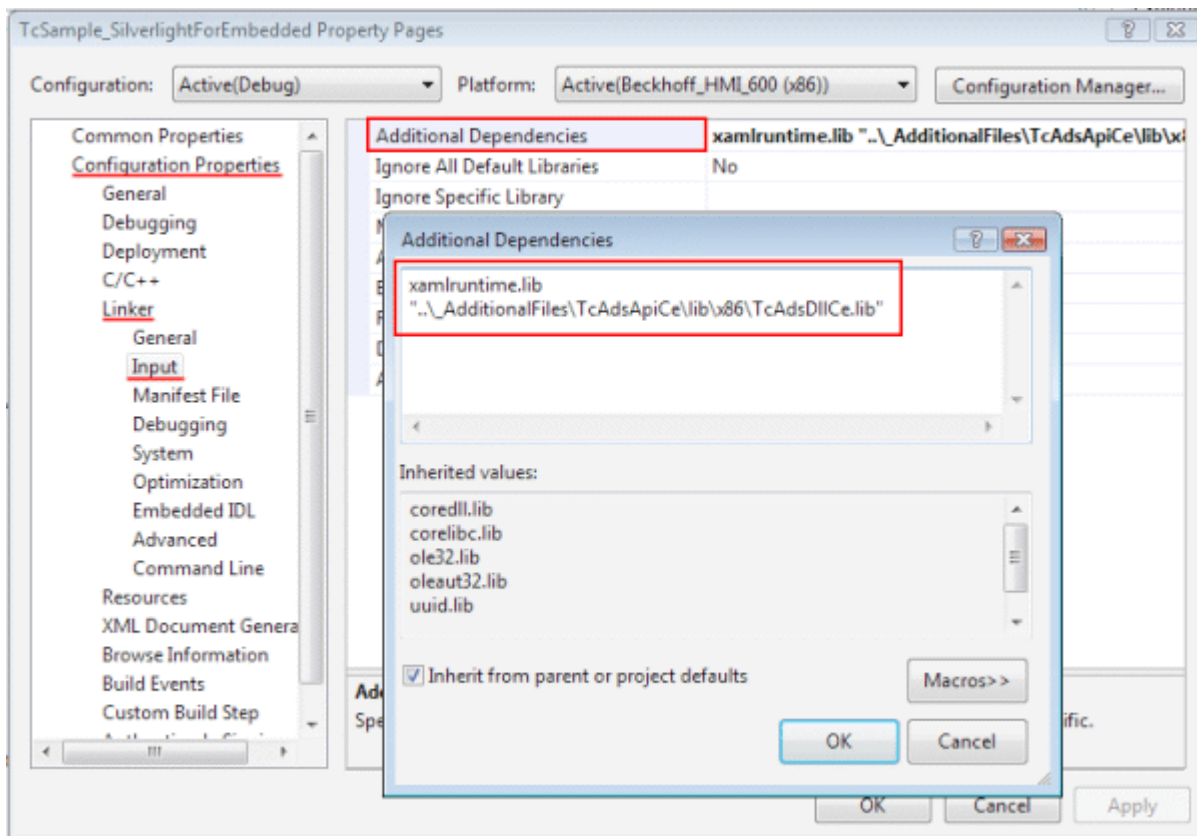
// cleanup
disconnect(port, &addr);
clickdelegate->Release();

return 0;
}

```

7. Properties

A connection to *xamlruntime.lib* and *TcAdsDIICe.lib* must be made in the project properties.



Download Silverlight for Windows Embedded sample

https://infosys.beckhoff.com/content/1033/tcsample_expression/Resources/12493869707.zip

i Sample for ARM devices

In the sample the X86 version of the TcAdsDllCe.lib is used. To build the sample for ARM devices, this library must be exchanged for the corresponding ARM version beforehand.

3.2 Sample Machine with Microsoft Silverlight and JavaScript

Microsoft Silverlight is a web presentation technology that can be displayed by all popular browsers (Internet Explorer 6/7, Mozilla Firefox, Apple Safari and Opera) through a corresponding plugin.

Target platforms

- Windows XP, XPE, WES
- Windows Vista
- Windows 7

Implementation

- JavaScript

Required software:

- **Runtime:**
 - Microsoft Silverlight 1.0
 - Microsoft Silverlight 1.1

Which runtime you need depends on whether you want to render a Silverlight 1.0 or 1.1 application in your browser.

- **Developer Tools:**

- Microsoft Visual Studio 2008 Beta 2
- Microsoft Silverlight Tools Alpha Refresh for Visual Studio (July 2007)

- or

- Microsoft Visual Studio 2005
- Microsoft Silverlight 1.0 Software Development Kit

Microsoft Visual Studio 2005 was used for this sample.

- **Designer Tools:**

- Expression Blend 2 August Preview

- **Others:**

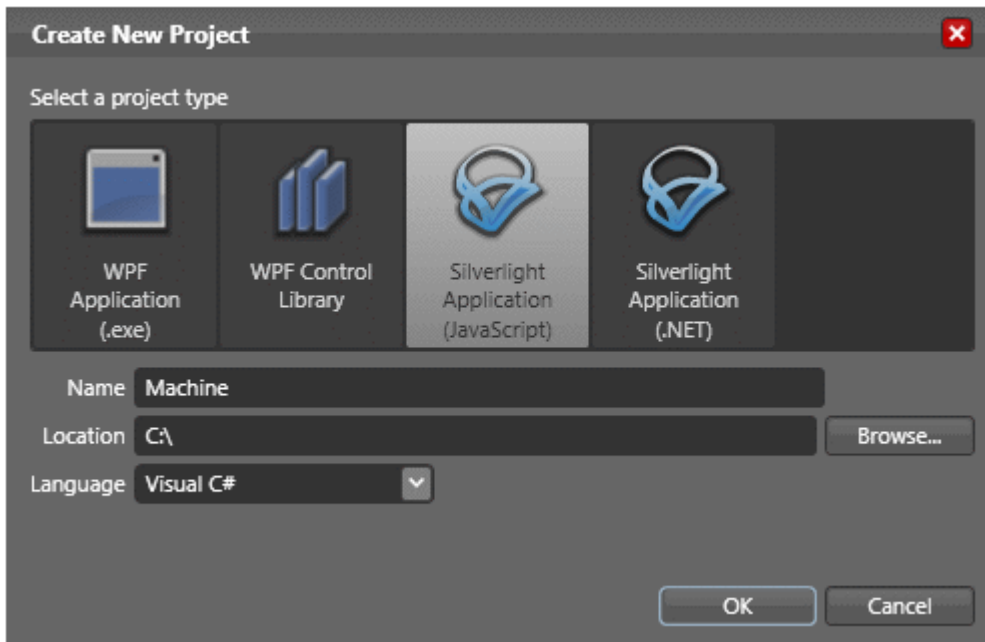
- TwinCAT 2.10
- Browser (e.g. Internet Explorer 7 or Mozilla Firefox)
- Microsoft .NET Framework Version 3.0

First steps...

Step by step you will learn how to develop a Silverlight application and how to integrate the TwinCAT ADS Web Service using an example.

1. Creating a new project:

Start Microsoft Expression Blend 2 and create a new interface via 'Menu → File → new Project...' . The 'Create New Project' dialog box opens and the type, name, location and programming language can be selected. In this sample, select the type 'Silverlight Application (JavaScript)' and the name 'Machine'.



2. Creating a user interface:

Only a few controls are available for creating the user interface. You can create a *RadioButton* with a *Textbox* and two *ellipses* packed into a *Canvas*. The *Progressbar* can be created with three *Rectangles*, which can be put into a separate *Canvas*.

The interface settings are stored in the *Page.xaml* file.

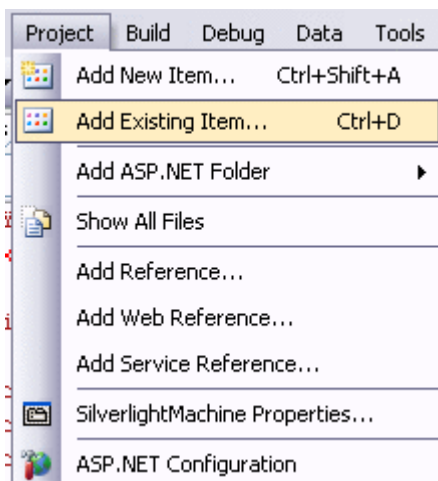
Note that you do not set the size to 'Auto' for any object. This can otherwise lead to errors later on.



In the upper left you see the two outputs that are also output to the Bus Terminals. The bottom left shows the variable for counting the workpieces. The cycle speed of the motor can be changed via the *Speed* field on the right. The *Steps* display shows the number of cycles that are output on output 1.

3. Add XMLHTTP.JS

In Visual Studio via 'Project → Add Existing Item...' add the file 'XMLHTTP.JS'. This file contains general methods for reading and writing PLC variables, as well as for converting data types.



4. Editing the source code

Open the Default.html file in Visual Studio and include the 'XMLHTTP.JS' file in the header.

```
<script type="text/javascript" src="xmlhttp.js"></script>
```

In the HTML page, add a JavaScript area in the HEAD section. The following source code must be inserted there:

First, the most important variables must be declared.

```
<script type="text/javascript"> //enter URL to webservice here: var url = "http://localhost/TcAdsWebService/TcAdsWebService.dll";
//enter netId here: var netId = "172.16.2.63.1.1";
//enter the port here: var port = 811;
//send soap request every x seconds: var refresh = 1000;

var inuse = false; var b64s, success, errors, req;
```



```
var vUp, vDown, vProgressbar, vCount, vFast, vSlow;
...
```

You have to adjust the URL of the TcAdsWebService as well as the NetID and the port accordingly.

The objects of the user interface cannot be easily accessed. For this to work, the objects are assigned to the variables already declared above after the application is started.

```
function Load(sender, EventArgs)
{
    vUp = sender.findName("pathUp");
    vDown = sender.findName("pathDown");
    vProgressbar = sender.findName("recProgressbar");
    vCount = sender.findName("txbCount");
    vFast = sender.findName("ellPointFast");
    vSlow = sender.findName("ellPointSlow");
}
```

The Load method does involve assigning the variables, but it is never called until now. This is done by changing the XAML code of the interface in Expression Blend 2. To do this, add *Loaded="Load"* to the topmost *canvas*.

```
<Canvasxmlns="http://schemas.microsoft.com/client/2007"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Width="368" Height="256"
    x:Name="Page"
    Loaded="Load"
>
...
```

```
function loop(x)
{
    Read(netId, port, '16416', '0', '86'); //send soap read request via xmlhttprequest
    window.setTimeout("loop("+x+")", x);
}
```

Reading a PLC variable

```
Read(netId, nPort, indexGroup, indexOffset, cbLen)
```

- **netId**: string that specifies the AMS Net ID on which the PLC can be found
- **nPort**: port number of the runtime system
- **indexGroup**: IndexGroup of the PLC variable
- **indexOffset**: first byte to be written to
- **ncbLen**: number of bytes to be written

```
function init()
{
    b64s = "ABCDEFGHGIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/-=";
    success = 0;
    errors = 0;
    loop(refresh); //send request every x seconds
}
```

Add *onload="init()"* to *<body>* so that the method is executed during loading.

```
<body onload="init()" >
```

The next function ensures that the values are read and output. It is important here that the address of the variable is correctly specified by Machine.pro for reading.

```
function processReqChange ()
{
    /*
    readyStates:
    0 = uninitialized
    1 = loading
    2 = loaded
    3 = interactive
    4 = complete
    */ if (req.readyState == 4)
    {
        // only if "OK" if (req.status == 200)
        {
            response = req.responseXML.documentElement;

            inuse = false;

            try//check if there was an error in the request
            {
                errortext = response.getElementsByTagName('faultstring')[0].firstChild.data;
                try
                {
                    errorcode = response.getElementsByTagName('errorcode')[0].firstChild.data;
                }
                catch (e){errorcode="-";}
                alert(errortext + " (" +errorcode+")");
                return;
            }
            catch (e)
            {
                errorcode=0;
            }

            var data;
            try//
            if the server returns a <ppData> element decode it, otherwise (write request) do nothing
            {
                data = response.getElementsByTagName('ppData')[0].firstChild.data;
                mode = "read";
            }
            catch (e)
            {
                data = "";
                mode = "";
            }

            if (mode=="read")
            {
                try
                {
                    data = b64t2d(data); //decode result string

                    steps = toInt(data.substr(1, 2));

                    bool = toInt(data.substr(5,2));
                    bool2 = toInt(data.substr(4,2));

                    count = toInt(data.substr(3, 2));

                    speed = toInt(data.substr(6, 2));
                }
                catch (e)
                {
                    alert("Parsing Failed:" + e);
                    return;
                }

                vProgressbar.Width = 306.321/100*steps*4;

                if (bool2 != "1")
                { vCount.Text = count.toString();}

                if (bool == "1")
```

```

        { vUp.Opacity=1.0;
          vDown.Opacity=0.0; }
    else if (bool2 == "1")
        { vUp.Opacity=0.0;
          vDown.Opacity=1.0; }
    else
        { vUp.Opacity=0.0;
          vDown.Opacity=0.0; }

    if (speed == "0")
        { vFast.Opacity=1.0;
          vSlow.Opacity=0.0; }
    else
        { vSlow.Opacity=1.0;
          vFast.Opacity=0.0; }

    }
}
else alert(req.statusText+" "+req.status); //cannot retrieve xml data
}
}
}

```

In the last two methods, the PLC variable used to control the speed of the machine is set to zero and one, respectively.

```

function Fast_MouseLeftButtonDown(sender, EventArgs)
{
    Write(netId, port, '16416', '6', '2', '0', 'int');
}
function Slow_MouseLeftButtonDown(sender, EventArgs)
{
    Write(netId, port, '16416', '6', '2', '1', 'int');
}

```

Writing PLC variables

```
Write(netId, port, indexGroup, indexOffset, cbLen, pwrData, type)
```

- **netId**: string that specifies the AMS Net ID on which the PLC can be found
- **nPort**: port number of the runtime system
- **indexGroup**: IndexGroup of the PLC variable
- **indexOffset**: first byte to be written to
- **ncbLen**: number of bytes to be written
- **pwrData**: array containing the data to be written
- **type**: "bool", "int" or "string"

As with the 'Load' function, you need to switch to Expression Blend 2 to make the two methods the 'click event' of their two buttons in the appropriate lines.

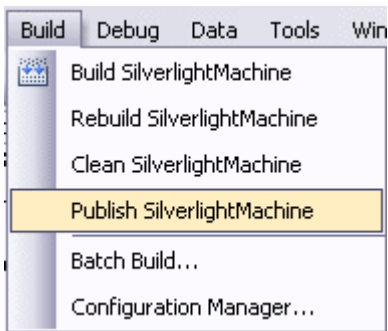
```

<Canvas x:Name="canvasFast" MouseLeftButtonDown="Fast_MouseLeftButtonDown" ...
<Canvas x:Name="canvasSlow" MouseLeftButtonDown="Slow_MouseLeftButtonDown" ...

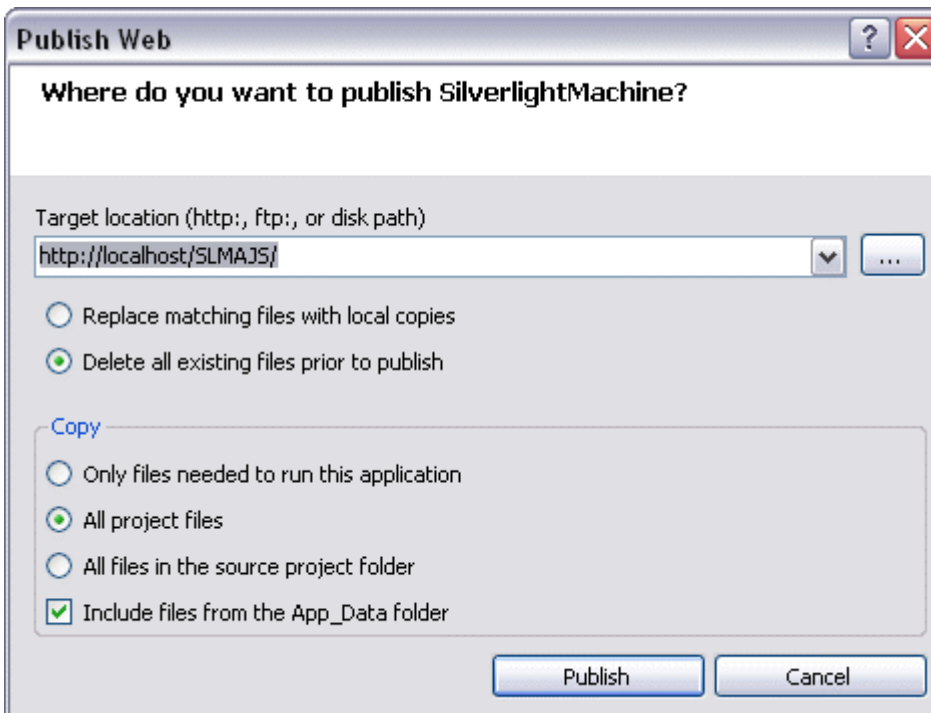
```

5. Testing:

First, debug your application. You will find that it does not work as you expect it to. Then go via 'Build → Publish'.



In the dialog box, select the 'Target location', and under 'Copy' select 'All project files'.



If the bottom left of the status bar says 'Publish succeeded', you can run and test your application in a browser.

Download:

https://infosys.beckhoff.com/content/1033/tcsample_expression/Resources/12493871115/.zip

4 Samples WPF

Windows Presentation Foundation

- target platforms: XP, XPE, WES, Vista, Win 7
- implementation: Visual C#, Visual Basic

Samples WPF

| Description | Sample |
|--|---|
| Sample 1: Machine WPF C# Sample [▶ 29] | https://infosys.beckhoff.com/content/1033/tcsample_expression/Resources/12493866891/.zip |
| Sample 2: Machine WPF Visual Basic Sample [▶ 36] | https://infosys.beckhoff.com/content/1033/tcsample_expression/Resources/12493868299/.zip |

4.1 Sample Machine with Microsoft Expression Blend (C#)

Microsoft Expression Blend is a program for creating program interfaces for C# and Visual Basic. In this sample an interface created with the program is linked with the Machine sample and subsequently integrated in the Vista Media Center. The programming language C# was used.

Target platform

- Windows Vista

Implementation

- Visual C#

Required software

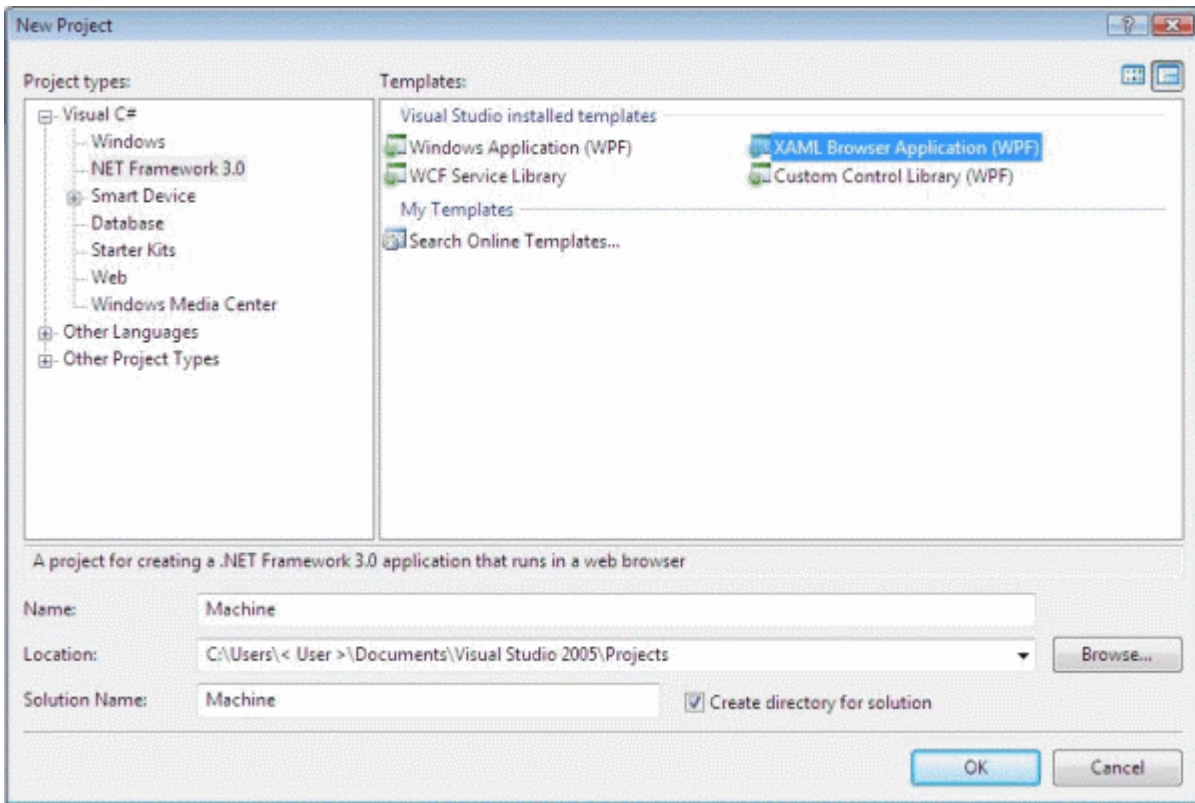
- Microsoft .NET Framework Version 3.0
- Microsoft Expression Blend
- Microsoft Visual Studio 2005
- Microsoft Windows Vista Media Center
- Microsoft Windows SDK for .Net Framework 3.0
- Microsoft Visual Studio 2005 extensions for .Net Framework 3.0 (November 2006 CTP)
- TwinCAT 2.10
- Notepad or other text editor

First steps ...

Step by step familiarization with the development of a program with Microsoft Visual Studio and Microsoft Expression Blend, integration of the TwinCAT ADS .NET component based on an example, and integration in the Vista Media Center.

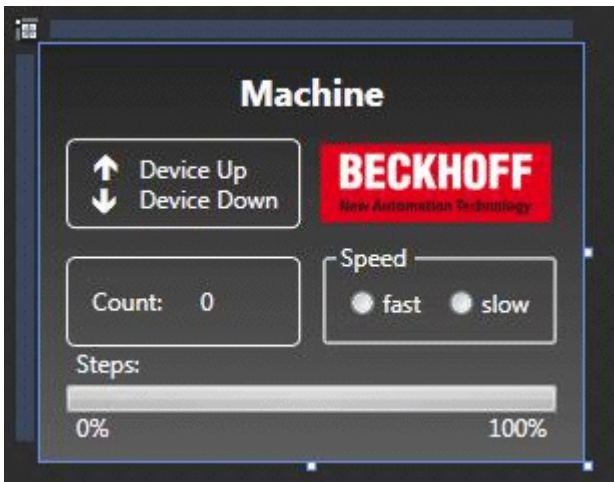
1. Creating a new project:

Start Microsoft Visual Studio and create new XAML browser application. Proceed via the menu 'File -> New -> Project...'. The dialog box 'New Project' opens. First select the project type: 'Project types -> Visual C# -> Net Framework 3.0'. The project type templates appear on the right. Select 'XAML Browser Application'. Enter a name for your project (in this case Machine) and specify the location.



2. Creating a user interface

Now change to Microsoft Expression Blend and open the project you just created in order to create the user interface.

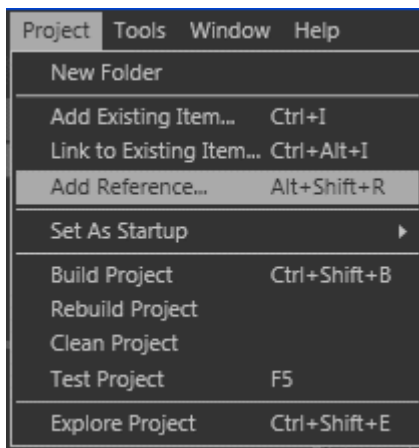


In the upper left you see the two outputs that are also output to the Bus Terminals. The bottom left shows the variable for counting the workpieces. The cycle speed of the motor can be changed via the 'Speed' field on the right. The 'Steps' display shows the number of cycles that are output on output 1.

To make the user interface constantly adapt its size, copy the upper grid and insert a view box instead of the grid. Now insert the grid into the view box. Now set the size of the page, the view box and the grid to 'Auto'. This may result in a shift of elements. You will then have to position them again. Please ensure that the size of the page, the view box and the grid is not inadvertently reset to fixed.

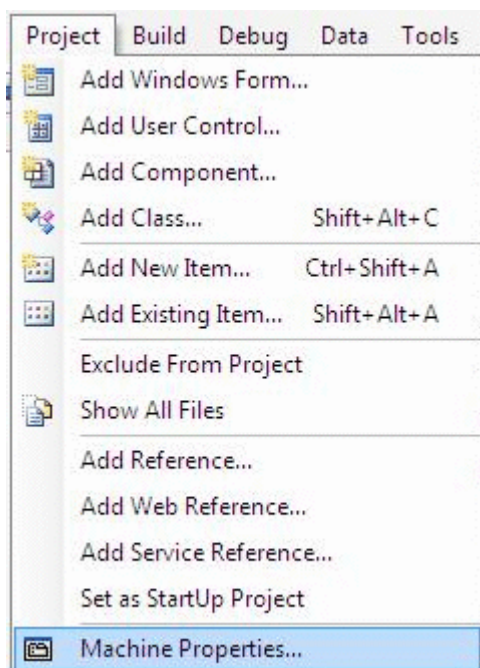
3. Adding a reference

Once the interface has been created, add a reference called 'TwinCAT.Ads.dll'. This can be done in Visual Studio or Expression Blend. In both cases proceed via the menu 'Project --> Add Reference'.

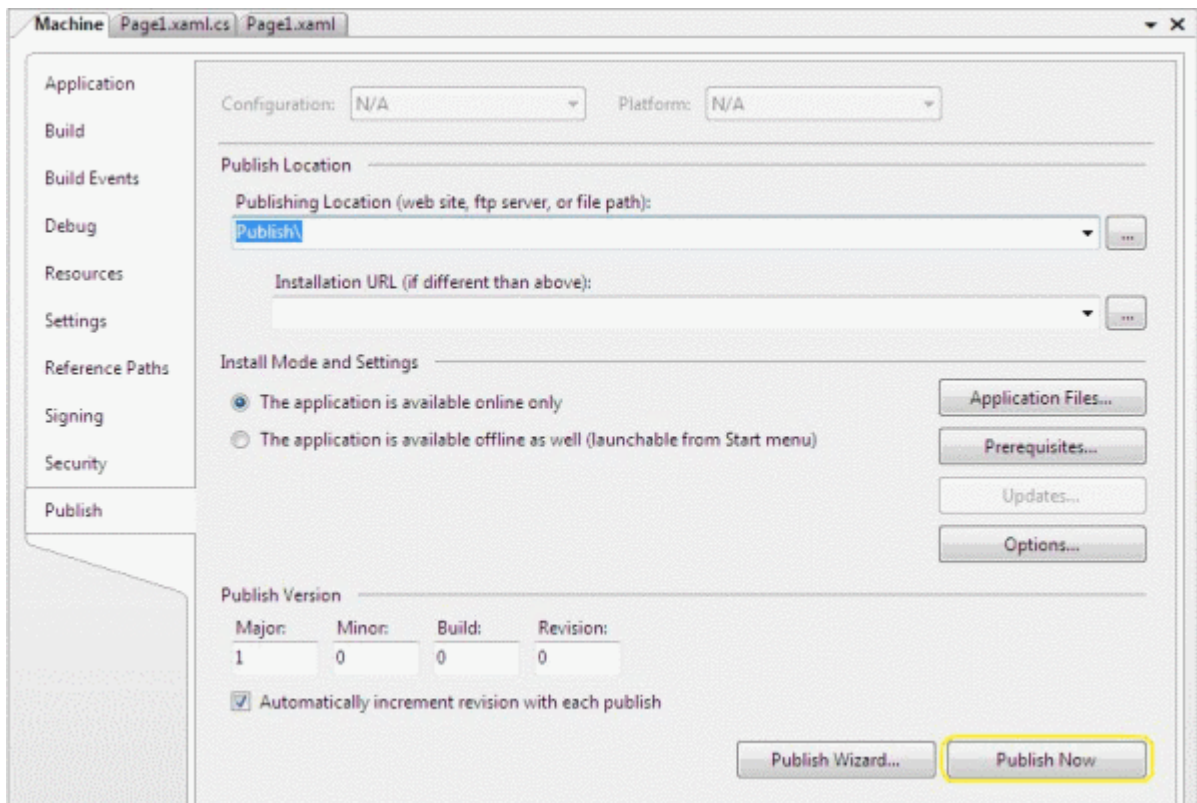


4. Security activation

Select 'Project -> <Project Name> Properties....' from the menu.



A tab opens, in which you can specify the project properties. Select 'Security' and then 'this is a full trust application'.



5. Editing the source code

Now the creation of the source code in C# can be started.

The required namespaces 'System.IO' and 'TwinCAT.Ads' are inserted into the top line of the source code.

```
using System.IO;
using TwinCAT.Ads;
```

This is followed by the declarations.

```
private TcAdsClient tcClient;
private AdsStream dataStream;
private BinaryReader binReader;
private int hEngine;
private int hDeviceUp;
private int hDeviceDown;
private int hSteps;
private int hCount;
private int hSwitchNotify;
private int hSwitchWrite;
```

The first method is the 'Load' method. It is used to generate instances of different classes and create a link to port 801.

```
//-----// Wird als erstes beim Starten des Programms
aufgerufen// Is activated first when the program is started//-----
-----private void Load(object sender, EventArgs e)
{
    try
    {
        // Eine neue Instanz der Klasse AdsStream erzeugen// Create an new instance of the AdsStream class
        dataStream = new AdsStream(7);

        // Eine neue Instanz der Klasse BinaryReader erzeugen// Create a new instance of the BinaryReader class
        binReader = new BinaryReader(dataStream);

        // Eine neue Instanz der Klasse TcAdsClient erzeugen// Create an new instance of the TcAdsClient class
        tcClient = new TcAdsClient();

        // Verbinden mit lokaler SPS - Laufzeit 1 - Port 801// Connecting to local PLC - Runtime 1 - Port 801
        tcClient.Connect(801);
    }
}
```



```

    }
    catch
    {
        MessageBox.Show("Fehler beim Laden");
    }
    //...

```

The variables in the 'Load' method are then linked and linked with a method (that still has to be written), which is called when a variable changes.

```

try
{
    // Initialisieren der Überwachung der SPS-
    Variablen// Initializing the monitoring of the PLC variables
    hEngine = tcClient.AddDeviceNotification(".engine", dataStream, 0, 1, AdsTransMode.OnChange, 10,
    0, null);
    hDeviceUp = tcClient.AddDeviceNotification(".deviceUp", dataStream, 1, 1, AdsTransMode.OnChange,
    10, 0, null);
    hDeviceDown = tcClient.AddDeviceNotification(".deviceDown", dataStream, 2, 1, AdsTransMode.OnCha
    nge, 10, 0, null);
    hSteps = tcClient.AddDeviceNotification(".steps", dataStream, 3, 1, AdsTransMode.OnChange, 10, 0
    , null);
    hCount = tcClient.AddDeviceNotification(".count", dataStream, 4, 2, AdsTransMode.OnChange, 10, 0
    , null);
    hSwitchNotify = tcClient.AddDeviceNotification(".switch", dataStream, 6, 1, AdsTransMode.OnChan
    ge, 10, 0, null);

    // Holen des Handles von 'switch' -
    wird für das Schreiben des Wertes benötigt// Getting the handle for 'switch' -
    needed for writing the value
    hSwitchWrite = tcClient.CreateVariableHandle(".switch");

    // Erstellen eines Events für Änderungen an den SPS-Variablen-
    Werten // Creating an event for changes of the PLC variable value
    tcClient.AdsNotification += newAdsNotificationEventHandler(tcClient_OnNotification);
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
}

```

6. Definition

Linking PLC variables:

The method AddDeviceNotification was used for linking the variables.

```

public int AddDeviceNotification(string variableName, AdsStream dataStream, int offset, int length,
    AdsTransMode transMode, int cycleTime, int maxDelay, object userData);

```

- **variableName:** name of the PLC variable.
- **dataStream:** data stream receiving the data.
- **offset:** interval in the data stream.
- **length:** length in the data stream.
- **transMode:** event if the variable changes.
- **cycleTime:** time (in ms) after which the PLC server checks whether the variable has changed.
- **maxDelay:** latest time (in ms) after which the event has finished.
- **userData:** object that can be used for storing certain data.

The method CreateVariableHandle was used for linking the variable 'hSwitchWrite'.

```

int TcAdsClient.CreateVariableHandle(string variableName);

```

- **variableName:** name of the PLC variable.

7. Writing the method:

A method that does not exist yet was referred to above. This method ('tcClient_OnNotification') is written next. The method is called if one of the PLC variables has changed.

```
//-----// wird bei Änderung einer SPS-
Variablen aufgerufen// is activated when a PLC variable changes//-----
-----private void tcClient_OnNotification(object sender, AdsNotificationEventArgs e)
{
    try
    {
        // Setzen der Position von e.DataStream auf die des aktuellen benötigten Wertes// Setting the po
        sition of e.DataStream to the position of the current needed value
        e.DataStream.Position = e.Offset;

        // Ermittlung welche Variable sich geändert hat// Detecting which variable has changedif(e.Notif
        icationHandle == hDeviceUp)
        {
            // Die Farben der Grafiken entsprechened der Variablen anpassen// Adapt colors of graphice a
            ccording to the variablesif (binReader.ReadBoolean() == true)
            {
                DeviceUp_LED.Foreground = newSolidColorBrush(Colors.Red);
            }
            else
            {
                DeviceUp_LED.Foreground = newSolidColorBrush(Colors.White);
            }
        }
        else if(e.NotificationHandle == hDeviceDown)
        {
            if (binReader.ReadBoolean() == true)
            {
                DeviceDown_LED.Foreground = newSolidColorBrush(Colors.Red);
            }
            else
            {
                DeviceDown_LED.Foreground = newSolidColorBrush(Colors.White);
            }
        }
        else if(e.NotificationHandle == hSteps)
        {
            // Einstellen der ProgressBar auf den aktuellen Schritt// Setting the ProgressBar to the cur
            rent step
            prgSteps.Value = binReader.ReadByte();
        }
        else if(e.NotificationHandle == hCount)
        {
            // Anzeigen des 'Zähler'-Wertes// Displaying the 'count' value
            lblCount.Text = binReader.ReadUInt16().ToString();
        }
        else if(e.NotificationHandle == hSwitchNotify)
        {
            // Markieren des korrekten RadioButtons// Checking the correct RadioButtonif (binReader.Read
            Boolean() == true)
            {
                optSpeedFast.IsChecked = true;
            }
            else
            {
                optSpeedSlow.IsChecked = true;
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
}
```

There are still two methods missing for setting the speed of the machine. They are used to switch a virtual switch by writing a value to the PLC variable 'switch'.

```
//-----// wird aufgerufen, wenn das Feld 'schnell'
markiert wird// is activated when the 'fast' field is marked//-----
-----private void optSpeedFast_Click(object sender, EventArgs e)
{
    try
```

```

    {
        tcClient.WriteAny(hSwitchWrite, true);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

//-----// wird aufgerufen, wenn das Feld 'langsam'
markiert wird// is activated when the 'slow' field is marked//-----
-----private void optSpeedSlow_Click(object sender, EventArgs e)
{
    try
    {
        tcClient.WriteAny(hSwitchWrite, false);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}
}

```

8. Deleting notifications and handles:

In the Close event of the window the links are enabled again with the method [DeleteDeviceNotification\(\)](#).

```

//-----// wird beim Beenden des Programms aufgerufe
n// is activated when ending the program//-----
private void Close(object sender, EventArgs e)
{
    try
    {
        // Löschen der Notifications und Handles// Deleting of the notification and handles
        tcClient.DeleteDeviceNotification(hEngine);
        tcClient.DeleteDeviceNotification(hDeviceUp);
        tcClient.DeleteDeviceNotification(hDeviceDown);
        tcClient.DeleteDeviceNotification(hSteps);
        tcClient.DeleteDeviceNotification(hCount);
        tcClient.DeleteDeviceNotification(hSwitchNotify);

        tcClient.DeleteVariableHandle(hSwitchWrite);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    tcClient.Dispose();
}
}

```

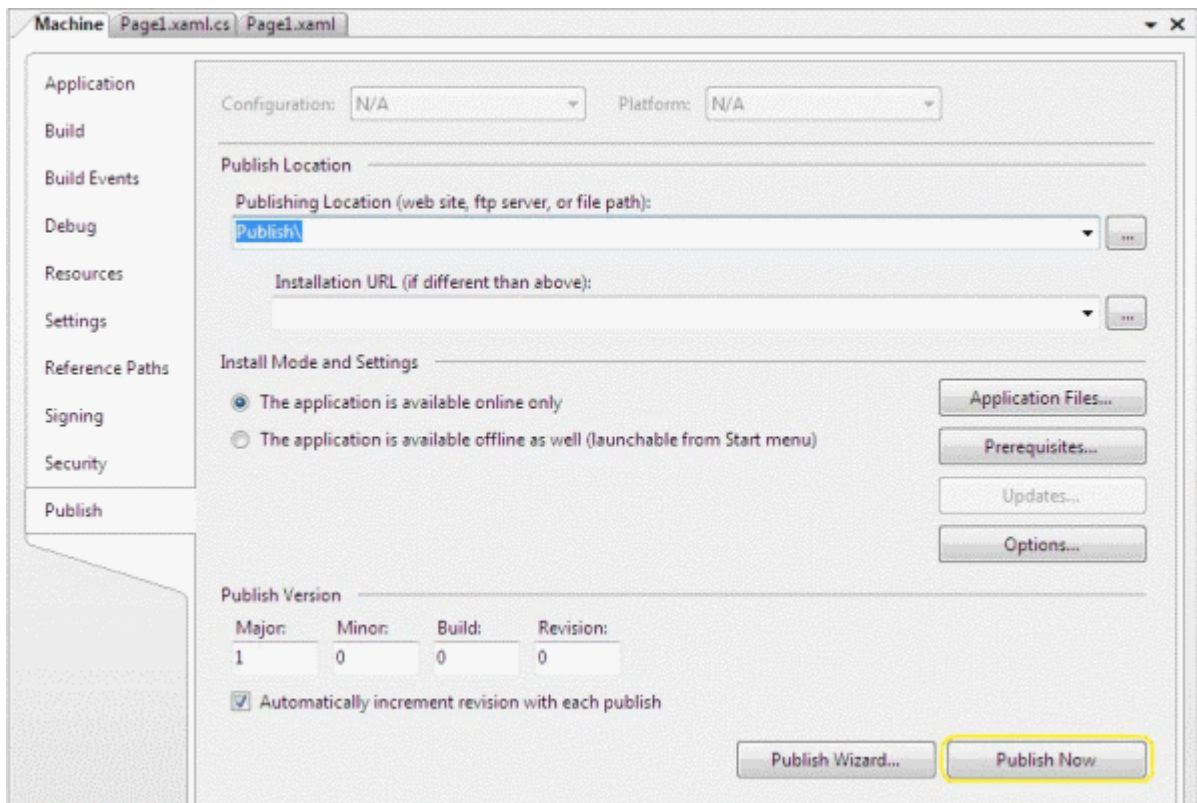
Last but not least we need to ensure that the methods are called for the right event. To do this, go to Expression Blend, select Page, switch to Events in Properties and enter 'Load' for 'Loaded' and 'Close' for 'Unloaded'.

The same must also be done with the two RadioButtons, only that here the event 'Click' is selected, as well as the method 'optSpeedFast_Click' or 'optSpeedSlow_Click'.

The PLC machine program Machine_Final.pro must run on runtime system 1 and the program can be tested in Internet Explorer 7.

9. Integration in Vista Media Center

If you have tested your project sufficiently and found no errors, you can now include it in the Media Center. Call up the project properties again in Visual Studio, but then go to 'Publish'. Click on 'Publish Now'. This creates an xbp file which you subsequently call up in the Media Center. This step is always required whenever the program has been modified and the change is to be transferred to the Media Center.



Now go to a text editor, e.g. Notepad, and enter the following:

```
<application
  URL = "C:
\Users\\Documents\Visual Studio 2005\Projects\Machine\Machine\Publish\Machine.xbap">
</application>
```

Save it to: 'C:\Users\\AppData\Roaming\Media Center Programs\Machine.mcl'. If you now start your Media Center you will find your program under 'Online Media -> program library -> programs by name -> Machine'.

This is the simplest form of integration into the Windows Vista Media Center. Further information on integration in the Media Center can be found [here](#).

10. Download Expression Blend sample:

https://infosys.beckhoff.com/content/1033/tcsample_expression/Resources/12493866891/.zip

4.2 Sample Machine with Microsoft Expression Blend (VB)

Microsoft Expression Blend is a program for creating program interfaces for C# and Visual Basic. In this sample an interface created with the program is linked with the Machine sample and subsequently integrated in the Vista Media Center. The Visual Basic programming language is used.

Target platform

- Windows Vista

Implementation

- Visual Basic

Required software

- Microsoft .NET Framework Version 3.0
- Microsoft Expression Blend, for further information click [here](#)
- Microsoft Visual Studio 2005

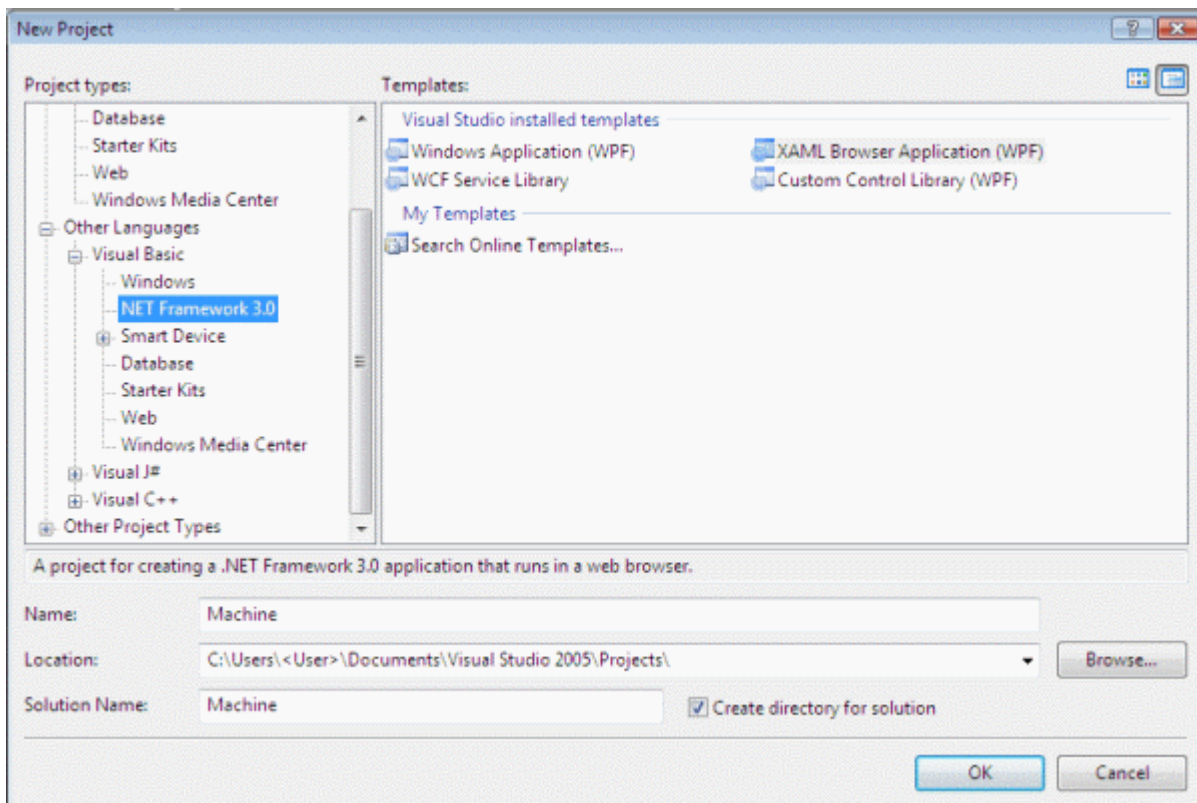
- Microsoft Windows Vista Media Center
- Microsoft Windows SDK for .Net Framework 3.0
- Microsoft Visual Studio 2005 extensions for .Net Framework 3.0 (November 2006 CTP)
- TwinCAT 2.10
- Notepad or other text editor

First steps ...

Step by step familiarization with the development of a program with Microsoft Visual Studio and Microsoft Expression Blend, integration of the TwinCAT ADS .NET component based on an example, and integration in the Vista Media Center.

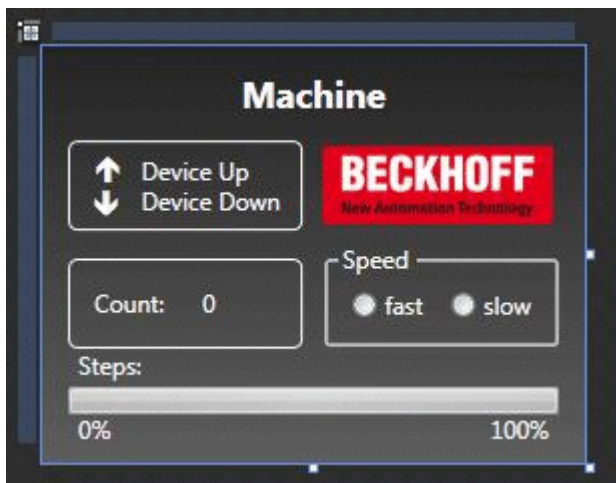
1. Creating a new project:

Start Microsoft Visual Studio and create new XAML browser application. Proceed via the menu 'File -> New -> Project...'. The dialog box 'New Project' opens. First select the project type: 'Project types -> Visual Basic -> Net Framework 3.0'. The project type templates appear on the right. Select 'XAML Browser Application'. Enter a name for your project (in this case 'Machine') and specify the location.



2. Creating a user interface

Now change to Microsoft Expression Blend and open the project you just created in order to create the user interface.

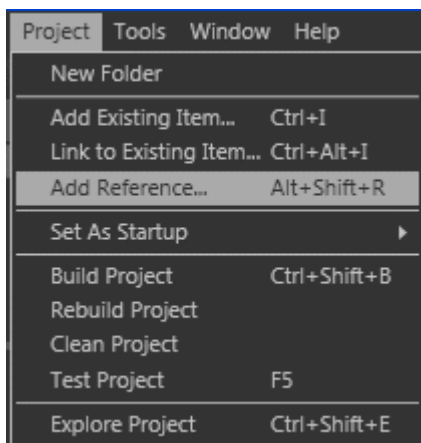


In the upper left you see the two outputs that are also output to the Bus Terminals. The bottom left shows the variable for counting the workpieces. The cycle speed of the motor can be changed via the 'Speed' field on the right. The 'Steps' display shows the number of cycles that are output on output 1.

To make the user interface constantly adapt its size, copy the upper grid and insert a view box instead of the grid. Now insert the grid into the view box. Now set the size of the page, the view box and the grid to 'Auto'. This may result in a shift of elements. You will then have to position them again. Please ensure that the size of the page, the view box and the grid is not reset to fixed.

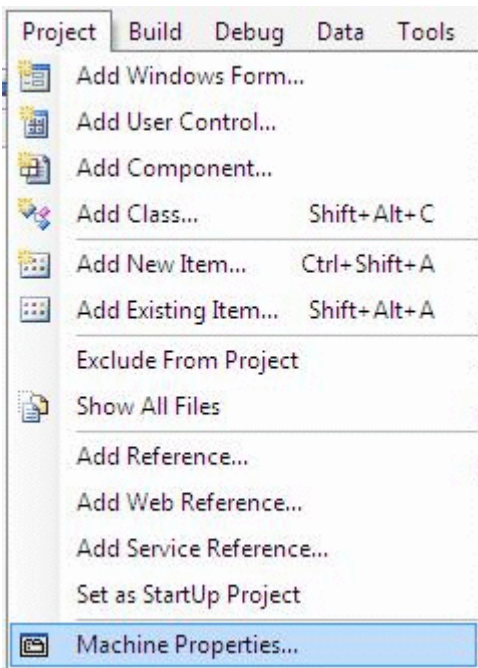
3. Adding a reference

Once the interface has been created, add a reference called 'TwinCAT.Ads.dll'. This can be done in Visual Studio or Expression Blend. In both cases proceed via the menu 'Project --> Add Reference'.

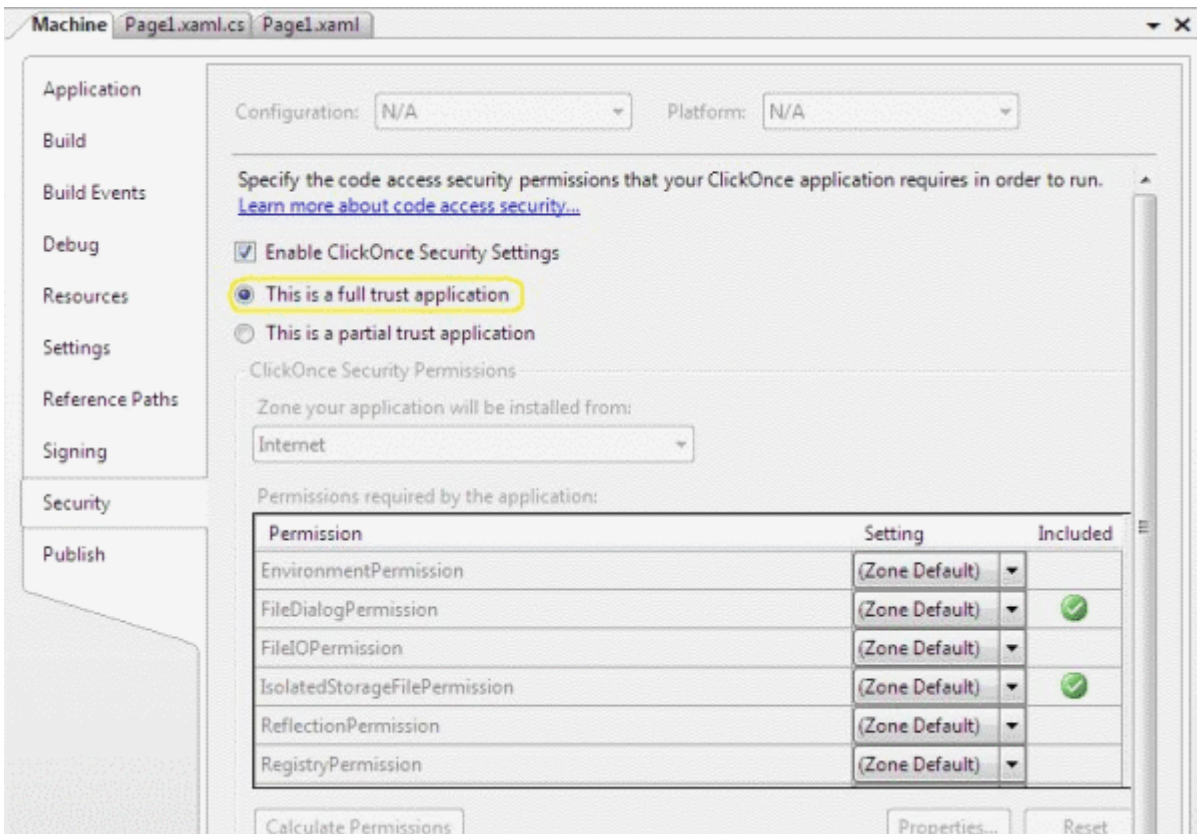


4. Security activation

Select 'Project -> <Project Name> Properties...!' from the menu.



A tab opens, in which you can specify the project properties. Select 'Security' and then 'this is a full trust application'.



5. Editing the source code

Now the creation of the source code in C# can be started. The required namespaces 'System.IO' and 'TwinCAT.Ads' are inserted into the top line of the source code.

```
Imports System.IO
Imports TwinCAT.Ads
```

This is followed by the declarations.

```

Private hEngine As Integer
Private hDeviceUp As Integer
Private hDeviceDown As Integer
Private hSteps As Integer
Private hCount As Integer
Private hSwitchNotify As Integer
Private hSwitchWrite As Integer

Private tcClient As TwinCAT.Ads.TcAdsClient
Private dataStream As TwinCAT.Ads.AdsStream
Private binReader As System.IO.BinaryReader

```

The first method is the 'Load' method. It is used to generate instances of different classes and create a link to port 801.

```

Private Sub Page1_Loaded(ByVal sender As Object, ByVal e As System.Windows.RoutedEventArgs) Handles Me.Loaded
    Try
        ' Eine neue Instanz der Klasse AdsStream erzeugen
        ' Create a new instance of the AdsStream class
        dataStream = New AdsStream(7)

        ' Eine neue Instanz der Klasse BinaryReader erzeugen
        ' Create a new instance of the BinaryReader class
        binReader = New BinaryReader(dataStream)

        ' Eine neue Instanz der Klasse TcAdsClient erzeugen
        ' Create a new instance of the TcAdsClient class
        tcClient = New TwinCAT.Ads.TcAdsClient()

        ' Verbinden mit lokaler SPS - Laufzeit 1 - Port 801
        ' Connecting to local PLC - Runtime 1 - Port 801
        tcClient.Connect(801)
    Catch
        MessageBox.Show("Error while loading")
    End Try
...

```

The variables in the 'Load' method are then linked and linked with a method (that still has to be written), which is called when a variable changes.

```

Try
    ' Initialisieren der Überwachung der SPS-Variablen
    ' Initializing the monitoring of the PLC variables
    hEngine = tcClient.AddDeviceNotification("engine", dataStream, 0, 1, AdsTransMode.OnChange, 10, 0, DBNull.Value)
    hDeviceUp = tcClient.AddDeviceNotification(".deviceUp", dataStream, 1, 1, AdsTransMode.OnChange, 10, 0, DBNull.Value)
    hDeviceDown = tcClient.AddDeviceNotification(".deviceDown", dataStream, 2, 1, AdsTransMode.OnChange, 10, 0, DBNull.Value)
    hSteps = tcClient.AddDeviceNotification(".steps", dataStream, 0, 1, AdsTransMode.OnChange, 10, 0, DBNull.Value)
    hCount = tcClient.AddDeviceNotification(".count", dataStream, 4, 2, AdsTransMode.OnChange, 10, 0, DBNull.Value)
    hSwitchNotify = tcClient.AddDeviceNotification(".switch", dataStream, 6, 1, AdsTransMode.OnChange, 10, 0, DBNull.Value)

    ' Holen des Handles von "switch" - wird für das Schreiben des Wertes benötigt
    ' Getting the handle for "switch" - needed for writing the value
    hSwitchWrite = tcClient.CreateVariableHandle(".switch")

    ' Erstellen eines Events für Änderungen an den SPS-Variablen-Werten
    ' Creating an event for changes of the PLC variable valuesAddHandler tcClient.AdsNotification, AddressOf tcClient_OnNotification
Catch
    MessageBox.Show("Error when connecting")
End Try
End Sub

```

6. Definition

Linking PLC variables:

The method AddDeviceNotification was used for linking the variables.

```

public int AddDeviceNotification(string variableName, AdsStream dataStream, int offset, int length, AdsTransMode transMode, int cycleTime, int maxDelay, object userData);

```


- **variableName:** name of the PLC variable.
- **dataStream:** data stream receiving the data.
- **offset:** interval in the data stream.
- **length:** length in the data stream.
- **transMode:** event if the variable changes.
- **cycletime:** time (in ms) after which the PLC server checks whether the variable has changed.
- **maxDelay:** latest time (in ms) after which the event has finished.
- **userData:** object that can be used for storing certain data.

The method `CreateVariableHandle` was used for linking the variable 'hSwitchWrite'.

```
int TcAdsClient.CreateVariableHandle(string variableName);
```

- **variableName:** name of the PLC variable.

7. Writing the method:

A method that does not exist yet was referred to above. This method ('tcClient_OnNotification') is written next. The method is called if one of the PLC variables has changed.

```
'-----' wird bei Änderung einer SPS-
Variablen aufgerufen' is activated when a PLC variable changes'-----
-----
Private Sub tcClient_OnNotification(ByVal sender As Object, ByVal e As AdsNotificationEventArgs)
    Try' Setzen der Position von e.DataStream auf die des aktuellen benötigten Wertes
        ' Setting the position of e.DataStream to the position of the current needed value
            e.DataStream.Position = e.Offset

            ' Ermittlung welche Variable sich geändert hat
            ' Detecting which variable has changedIf (e.NotificationHandle = hDeviceUp) Then'Die Farben
der Grafiken entsprechen der Variablen anpassen
            'Adapt colors of graphics according to the variablesIf (binReader.ReadBoolean() = True) Then
                DeviceUp_LED.Foreground = New SolidColorBrush(Colors.Red)
            Else
                DeviceUp_LED.Foreground = New SolidColorBrush(Colors.White)
            End If
            ElseIf (e.NotificationHandle = hDeviceDown) Then
                If (binReader.ReadBoolean() = True) Then
                    DeviceDown_LED.Foreground = New SolidColorBrush(Colors.Red)
                Else
                    DeviceDown_LED.Foreground = New SolidColorBrush(Colors.White)
                End If
            ElseIf (e.NotificationHandle = hSteps) Then' Einstellen der ProgressBar auf den aktuellen Sc
hritt
                ' Setting the ProgressBar to the current step
                prgSteps.Value = (binReader.ReadByte() * 4)
            ElseIf (e.NotificationHandle = hCount) Then' Anzeigen des "count"-Wertes
                ' Displaying the "count" value
                lblCount.Content = binReader.ReadUInt16().ToString()
            ElseIf (e.NotificationHandle = hSwitchNotify) Then' Markieren des korrekten RadioButtons
                ' Checking the correct RadioButtonIf (binReader.ReadBoolean() = True) Then
                    optSpeedFast.IsChecked = TrueElse
                    optSpeedSlow.IsChecked = True
                End If
            End If
        Catch
            MessageBox.Show("Error")
        End Try
    End Sub
```

There are still two methods missing for setting the speed of the machine. They are used to switch a virtual switch by writing a value to the PLC variable 'switch'.

```
'-----' wird aufgerufen, wenn das Feld 'schnell' ma
rkiert wird' is activated when the 'fast' field is marked'-----
-----
Private Sub optSpeedFast_Click(ByVal sender As Object, ByVal e As System.Windows.RoutedEventArgs) Ha
ndles optSpeedFast.Click
```

```

    Try
        tcClient.WriteAny(hSwitchWrite, True)
    Catch
        MessageBox.Show("Error")
    End Try
End Sub'-----' wird aufgerufen, wenn das Feld 'lang
sam' markiert wird' is activated when the 'slow' field is marked'-----
-----
Private Sub optSpeedSlow_Click(ByVal sender As Object, ByVal e As System.Windows.RoutedEventArgs) Handles optSpeedSlow.Click
    Try
        tcClient.WriteAny(hSwitchWrite, False)
    Catch
        MessageBox.Show("Error")
    End Try
End Sub

```

8. Deleting notifications and handles:

In the Close event of the window the links are enabled again with the method [DeleteDeviceNotification\(\)](#).

```

//-----// wird beim Beenden des Programms aufgerufe
n// is activated when ending the program//-----
Private Sub Page1_Unloaded(ByVal sender As Object, ByVal e As System.Windows.RoutedEventArgs) Handles Me.Unloaded
    Try' Löschen der Notifications und Handles
        ' Deleting of the notifications and handles
        tcClient.DeleteDeviceNotification(hEngine)
        tcClient.DeleteDeviceNotification(hDeviceUp)
        tcClient.DeleteDeviceNotification(hDeviceDown)
        tcClient.DeleteDeviceNotification(hSteps)
        tcClient.DeleteDeviceNotification(hCount)
        tcClient.DeleteDeviceNotification(hSwitchNotify)

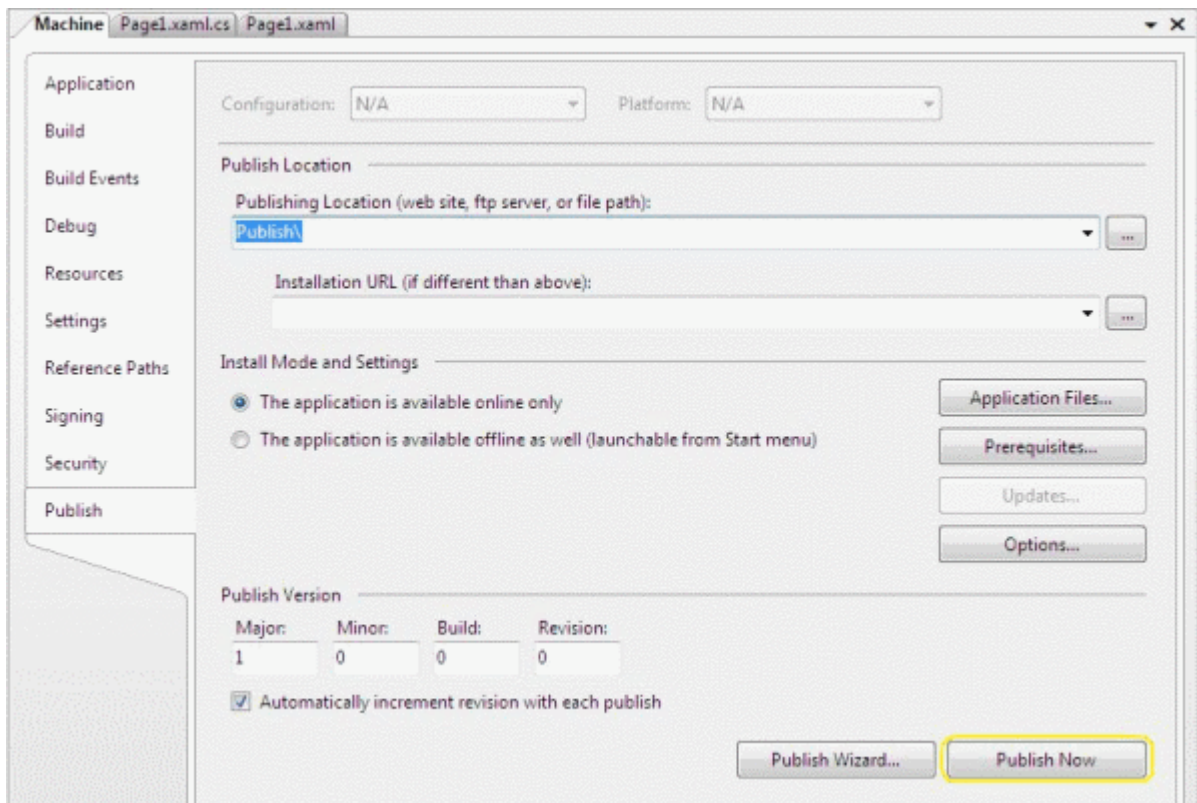
        tcClient.DeleteVariableHandle(hSwitchWrite)
    Catch
        MessageBox.Show("Error")
    End Try
    tcClient.Dispose()
End Sub

```

The Machine_Final.pro PLC machine program must run in runtime system 1, and the program can be tested in Internet Explorer 7.

9. Integration in Vista Media Center

If you have tested your project sufficiently and found no errors, you can now include it in the Media Center. Call up the project properties again in Visual Studio, but then go to 'Publish'. Click on 'Publish Now'. This creates an xbp file which you subsequently call up in the Media Center. This step is always required whenever the program has been modified and the change is to be transferred to the Media Center.



Now go to a text editor, e.g. Notepad, and enter the following:

```
<application
  URL = "C:
\Users\\Documents\Visual Studio 2005\Projects\Machine\Machine\Publish\Machine.xbap">
</application>
```

Save it to: 'C:\Users\\AppData\Roaming\Media Center Programs\Machine.mcl'. If you now start your Media Center you will find your program under 'Online Media -> program library -> programs by name -> Machine'.

This is the simplest form of integration into the Windows Vista Media Center. Further information on integration in the Media Center can be found [here](#).

10. Download Expression Blend sample:

https://infosys.beckhoff.com/content/1033/tcsample_expression/Resources/12493868299/.zip

5 ADS Return Codes

Grouping of error codes:

Global error codes: [ADS Return Codes \[▶ 44\]](#)... (0x9811_0000 ...)

Router error codes: [ADS Return Codes \[▶ 44\]](#)... (0x9811_0500 ...)

General ADS errors: [ADS Return Codes \[▶ 45\]](#)... (0x9811_0700 ...)

RTime error codes: [ADS Return Codes \[▶ 46\]](#)... (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_INVALIDAMSFAGMENT | Invalid AMS fragment. |
| 0x1D | 29 | 0x9811001D | ERR_TLSEND | 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_LICENSEDEMOMDENIED | 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_INVALIDPARG | 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_SYNC TIMEOUT | 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_NOMORESVM | 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_IRQNOTLESSOREQUAL | 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/tx1000

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com

