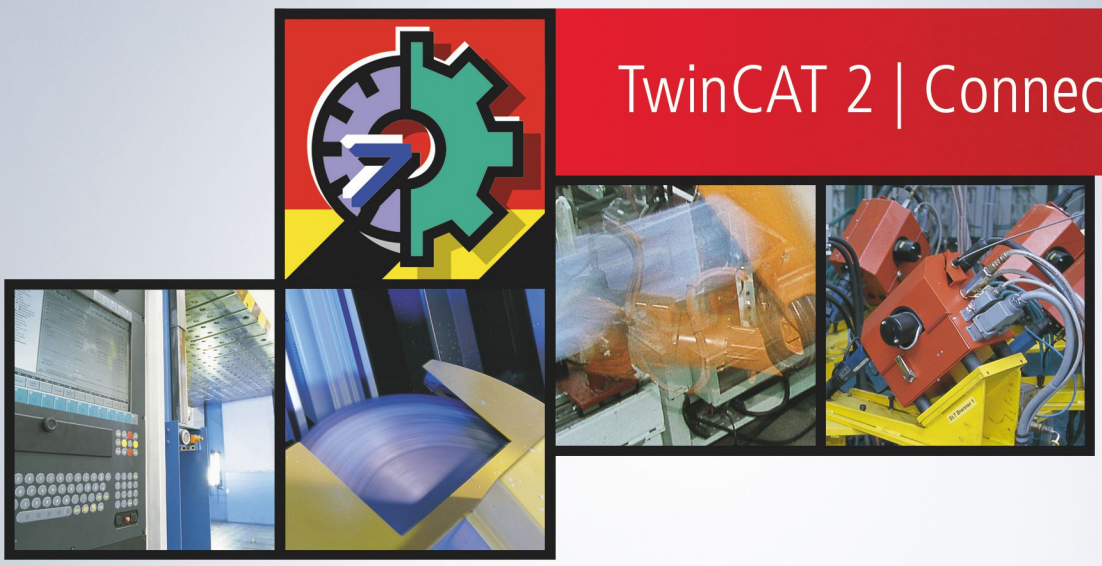
BECKHOFF New Automation Technology

Handbuch | DE

TX1000

TwinCAT 2 | ADS Silverlight/Expression



11.08.2023 | Version: 1.2

1 Vorwort

1.1 Hinweise zur Dokumentation

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

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

Disclaimer

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

Marken

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

Patente

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

EtherCAT®

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

Copyright

1.2 Zu Ihrer Sicherheit

Sicherheitsbestimmungen

Lesen Sie die folgenden Erklärungen zu Ihrer Sicherheit.
Beachten und befolgen Sie stets produktspezifische Sicherheitshinweise, die Sie gegebenenfalls an den entsprechenden Stellen in diesem Dokument vorfinden.

Haftungsausschluss

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

Qualifikation des Personals

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

Signalwörter

Im Folgenden werden die Signalwörter eingeordnet, die in der Dokumentation verwendet werden. Um Personen- und Sachschäden zu vermeiden, lesen und befolgen Sie die Sicherheits- und Warnhinweise.

Warnungen vor Personenschäden

⚠ GEFAHR

Es besteht eine Gefährdung mit hohem Risikograd, die den Tod oder eine schwere Verletzung zur Folge hat.

⚠ WARNUNG

Es besteht eine Gefährdung mit mittlerem Risikograd, die den Tod oder eine schwere Verletzung zur Folge haben kann.

⚠ VORSICHT

Es besteht eine Gefährdung mit geringem Risikograd, die eine mittelschwere oder leichte Verletzung zur Folge haben kann.

Warnung vor Umwelt- oder Sachschäden

HINWEIS

Es besteht eine mögliche Schädigung für Umwelt, Geräte oder Daten.

Information zum Umgang mit dem Produkt

Diese Information beinhaltet z. B.:
Handlungsempfehlungen, Hilfestellungen oder weiterführende Informationen zum Produkt.

1.3 Hinweise zur Informationssicherheit

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

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

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

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

2 Übersicht

Silverlight

Silverlight for Embedded
- Zielplattform: Windows CE 6 R3
- Implementierung: C++

Silverlight
- Zielplattformen: XP, XPE, WES, Vista, Win 7
- Implementierung: JavaScript, Visual C#

Beispiele Silverlight

Beschreibung	Beispiel
Beispiel 1: Machine Silverlight for Embedded C++ Sample [▶ 7]	https://infosys.beckhoff.com/content/1031/tcsample_expression/Resources/12493869707.zip
Beispiel 2: Machine Siverlight JavaScript Sample [▶ 22]	https://infosys.beckhoff.com/content/1031/tcsample_expression/Resources/12493871115.zip

Windows Presentation Foundation

- Zielplattformen: XP, XPE, WES, Vista, Win 7
- Implementierung: Visual C#, Visual Basic

Beispiele WPF

Beschreibung	Beispiel
Beispiel 1: Machine WPF C# Sample [▶ 29]	https://infosys.beckhoff.com/content/1031/tcsample_expression/Resources/12493866891.zip
Beispiel 2: Machine WPF Visual Basic Sample [▶ 36]	https://infosys.beckhoff.com/content/1031/tcsample_expression/Resources/12493868299.zip

Dokumente hierzu

- sampleexpressionvista.zip (Resources/zip/12493866891.zip)
- sampleexpressionvistavb.zip (Resources/zip/12493868299.zip)

3 Beispiele Silverlight

Silverlight

Silverlight for Embedded
- Zielplattform: Windows CE 6 R3
- Implementierung: C++

Silverlight
- Zielplattformen: XP, XPE, WES, Vista, Win 7
- Implementierung: JavaScript, Visual C#

Beispiele Silverlight

Beschreibung	Beispiel
Beispiel 1: Machine Silverlight for Embedded C++ Sample [▶ 7]	https://infosys.beckhoff.com/content/1031/tcsample_expression/Resources/12493869707.zip
Beispiel 2: Machine Siverlight JavaScript Sample [▶ 22]	https://infosys.beckhoff.com/content/1031/tcsample_expression/Resources/12493871115.zip

3.1 Beispiel Maschine mit Microsoft Silverlight for Windows Embedded

Eine Neuerung in Windows Embedded CE 6.0 R3 ist Silverlight for Windows Embedded. Mit dieser neuen Technologie können Bedienoberflächen von CE Geräten nun in XAML beschrieben werden und mit Tools wie Microsoft Expression Blend gestaltet werden. Anhand des Maschine Beispiels wird hier die Erstellung einer Silverlight for Windows Embedded Applikation mit Einbindung der ADS Komponente beschrieben.

Zielplattform

- Windows CE 6 R3

Implementierung

- C++

Erforderliche Software

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

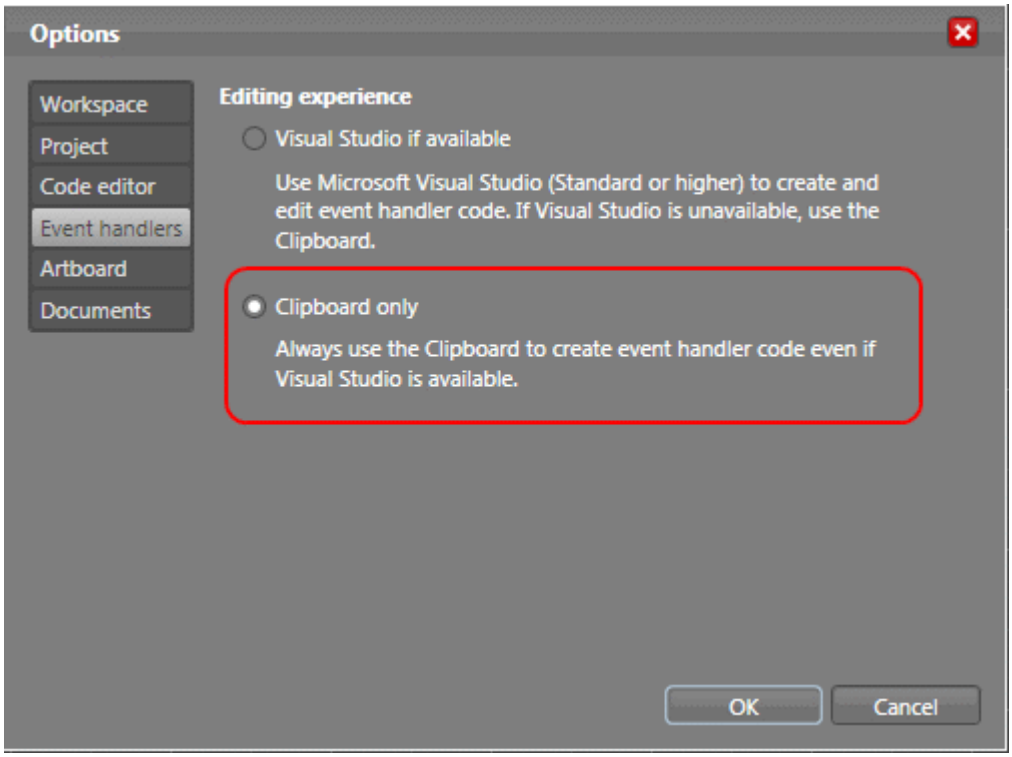
Erforderliche Hardware

- Windows CE 6.0 R3 Gerät (z.B. CX1020)

Die ersten Schritte...

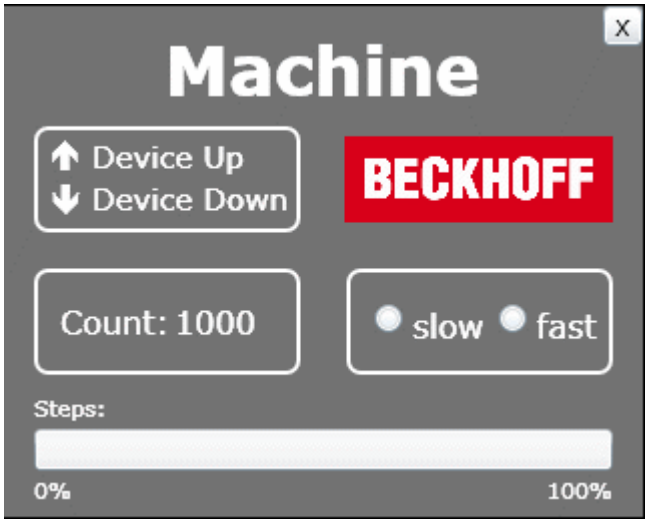
1. Neues Silverlight 2 Projekt erstellen:
Das Design einer Silverlight for Windows Embedded Applikation wird in XAML beschrieben. Dazu wird mit Microsoft Expression Blend 2 SP1 über *'File -> New Projekt'* ein Silverlight 2 Projekt erstellt. Die dabei erstellte Visual Studio Solution wird in diesem Beispiel nicht benötigt. Zudem kann die Auswahl der Programmiersprache (Visual C# oder Visual Basic) außer Acht gelassen werden. Silverlight for Windows Embedded unterstützt nur Visual C++, welches nicht in Expression Blend integriert ist. Es ist daher auch

nicht möglich den Quellcode zu verwenden, der von diesem Tool generiert wird. Daektivieren Sie die Visual Studio Integration in Expression Blend um eine unnötige automatische Generation von Visual C# und Visual Basic Code zu vermieden. Wählen Sie dazu: *'Options->Event handlers' 'Clipboard only'.*



2. Bedienoberfläche erstellen

In Expression Blend kann nun die Bedienoberfläche erstellt werden.



Im oberen linken Bereich sind die beiden Ausgänge zu sehen, die auch auf den Busklemmen ausgegeben werden. Unten links ist die Variable abgebildet, welche die Werkstücke zählt. Rechts kann die Geschwindigkeit eingestellt werden. Die Anzeige 'Steps' entspricht der Anzahl der Takte. Oben rechts wurde zudem noch ein Button zum Beenden des Programms erstellt.

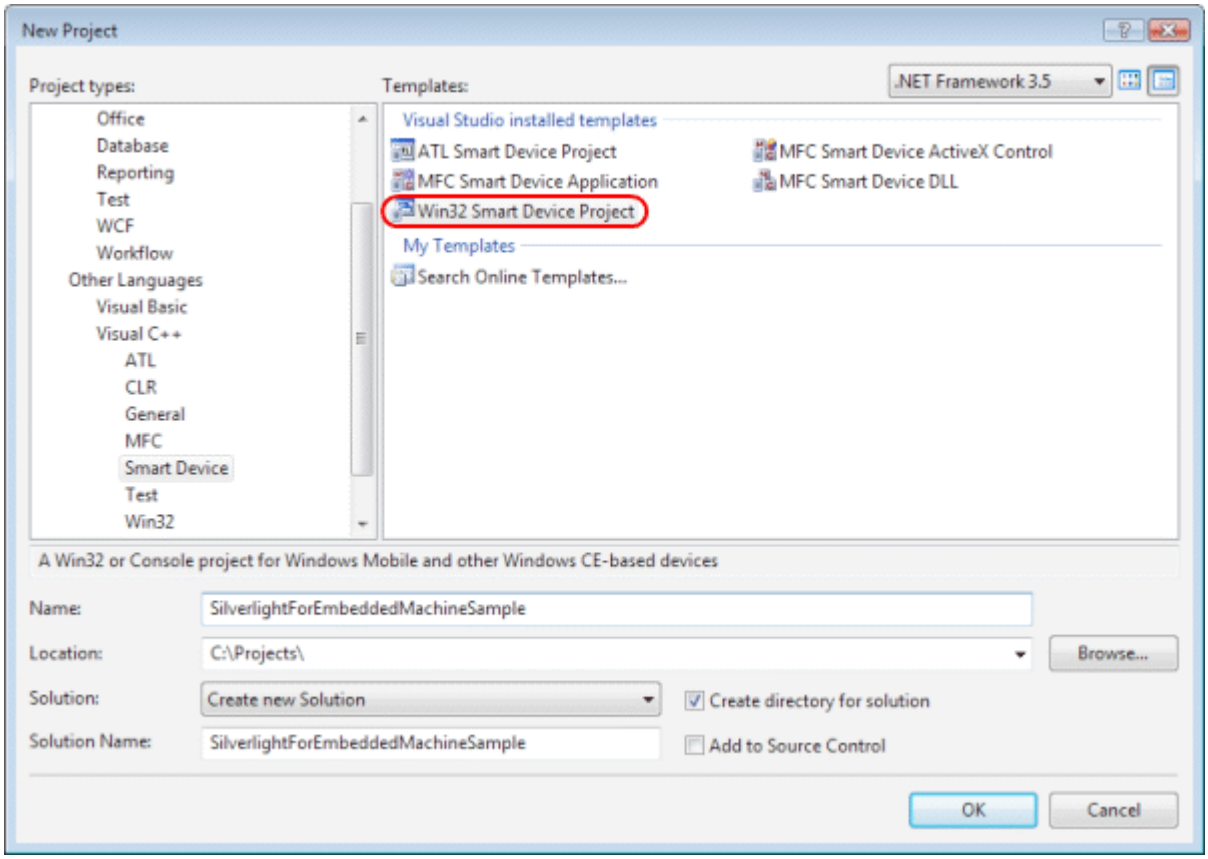
```
<UserControlxmlns="http: schemas.microsoft.com winfx 2006 xaml presentation" xmlns:x="http://
schemas.microsoft.com/winfx/2006/xaml" Width="319" Height="255">
    <!-- Timelines -->
    <UserControl.Resources>
        <!-- Timeline Device Down -->
        <Storyboardx:Name="timelineDeviceDown">
            <ColorAnimationUsingKeyFramesBeginTime="00:00:00" Storyboard.TargetName="txtDeviceDown"
Storyboard.TargetProperty="(TextBlock.Foreground).(SolidColorBrush.Color)">
                <SplineColorKeyFrameKeyTime="00:00:00.4000000" Value="#FFFF0000"/>
```

```
            </ColorAnimationUsingKeyFrames>
        </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="#FFFF0000"/>
            </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="#FFFFFFFF" FontSize="34"/>
        <!-- Device Up / Device Down -->
        <GridMargin="15,60,0,0" HorizontalAlignment="Left" VerticalAlignment="Top" Height="53"
Width="132.532">
            <RectangleFill="{x:Null}" Stroke="#FFFFFFFF" StrokeThickness="2" RadiusX="6"
RadiusY="6"/>
            <TextBlockText="Device Up" Margin="28.823,5.396,-8.823,0" VerticalAlignment="Top"
Foreground="#FFFFFFFF" FontSize="16"/>
            <TextBlockText="Device Down" Margin="29.002,0,-9.002,4.994" VerticalAlignment="Bottom"
Foreground="#FFFFFFFF" FontSize="16"/>
            <TextBlockx:Name="txtDeviceDown" Text="ê" Margin="7.517,0,0,4.998"
HorizontalAlignment="Left" VerticalAlignment="Bottom" FontFamily="Wingdings" FontWeight="Bold"
Foreground="#FFFFFFFF" FontSize="16"/>
            <TextBlockx:Name="txtDeviceUp" Text="é" Margin="7.517,5.497,0,0"
HorizontalAlignment="Left" VerticalAlignment="Top" FontFamily="Wingdings" FontWeight="Bold"
Foreground="#FFFFFFFF" FontSize="16"/>
        </Grid>
        <!-- Counter -->
        <GridMargin="15,0,0,71" HorizontalAlignment="Left" VerticalAlignment="Bottom" Height="53"
Width="133">
            <RectangleFill="{x:Null}" Stroke="#FFFFFFFF" StrokeThickness="2" RadiusX="6"
RadiusY="6"/>
            <StackPanelMargin="12.991,16,0,16" HorizontalAlignment="Left" Orientation="Horizontal"
Width="113">
                <TextBlockText="Count:" Width="54.824" Foreground="#FFFFFFFF" FontSize="16"/>
                <TextBlockx:Name="txtCount" Margin="2,0,0,0" Foreground="#FFFFFFFF" FontSize="16"/>
            </StackPanel>
        </Grid>
        <!-- Speed -->
        <GridMargin="0,0,15,71" Height="53" HorizontalAlignment="Right" VerticalAlignment="Bottom"
Width="132.532">
            <RectangleFill="{x:Null}" Stroke="#FFFFFFFF" 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="#FFFFFFFF" FontSize="16" Height="19.496"/>
                <RadioButtonx:Name="radSpeedFast" Content="fast" Foreground="#FFFFFFFF"
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="#FFFFFFFF"/>
            <TextBlockText="0%" HorizontalAlignment="Left" VerticalAlignment="Bottom"
Foreground="#FFFFFFFF"/>
            <TextBlockText="100%" HorizontalAlignment="Right" VerticalAlignment="Bottom"
Foreground="#FFFFFFFF"/>
        </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>
```

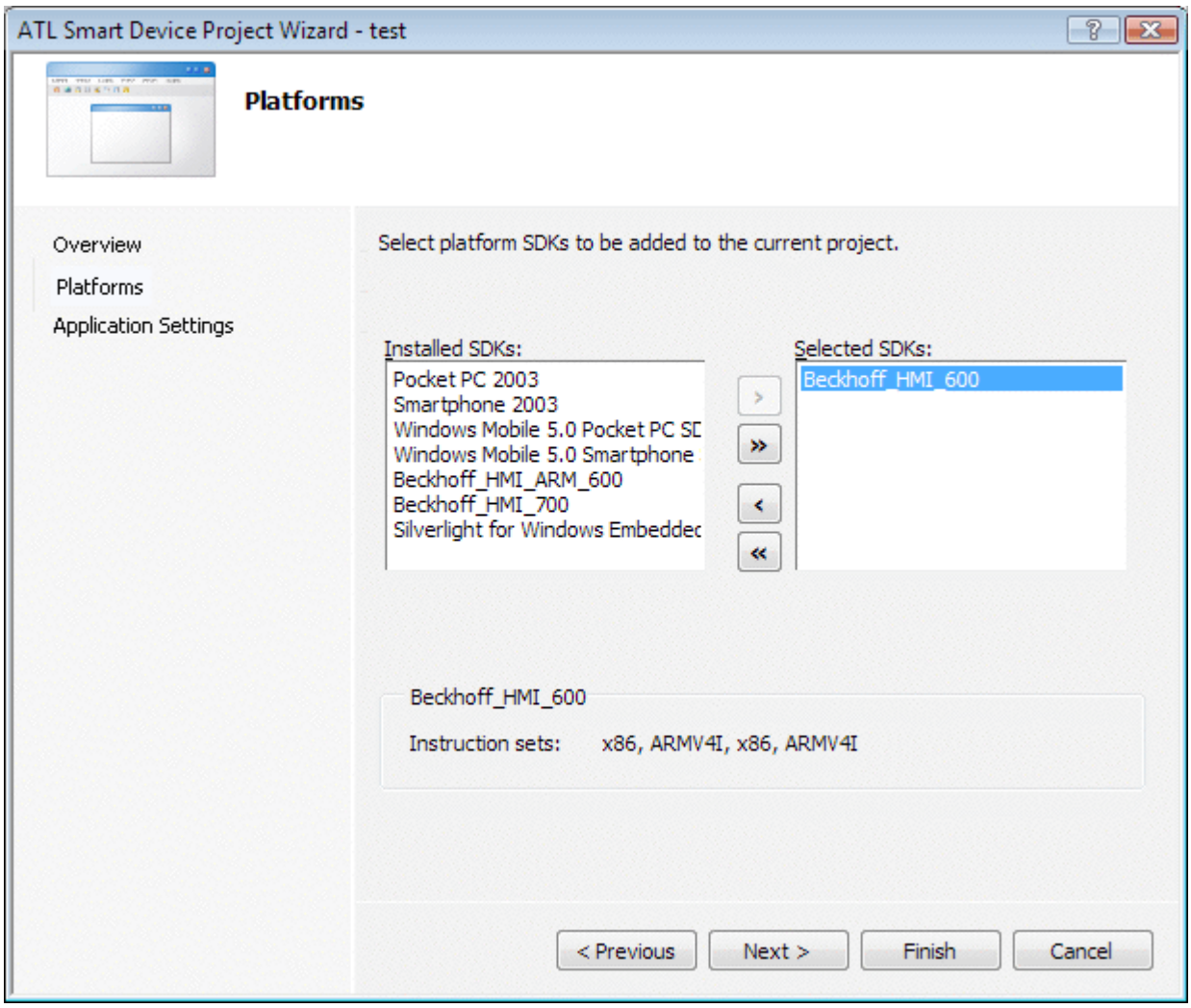
3. Neues Win32 Smart Device Projekt erstellen

In Visual Studio 2008 muss nun ein neues Win32 Smart Device Projekt erstellt werden.

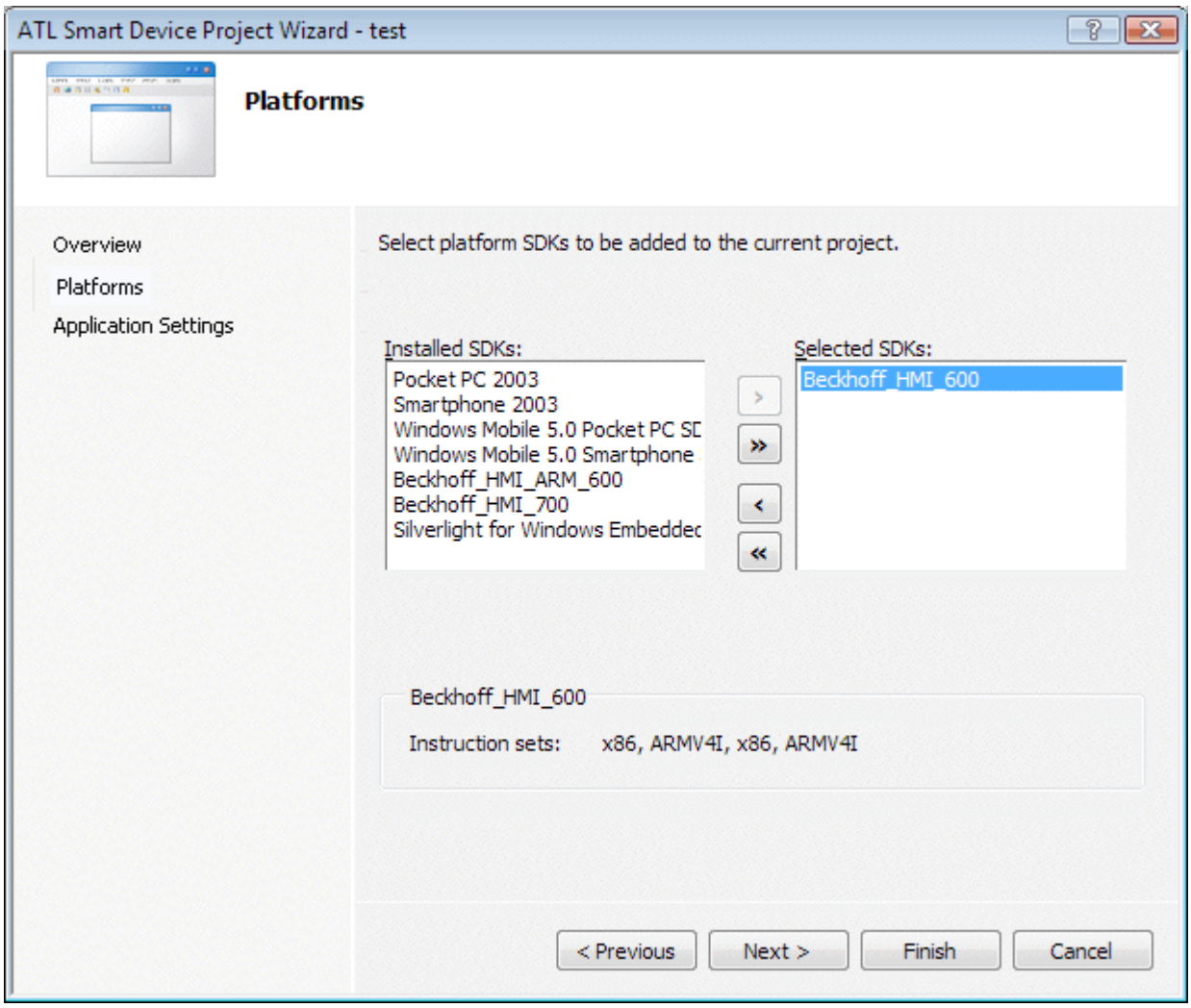
Sollte das Beckhoff HMI 600 SDK noch nicht auf dem Rechner installiert sein, so installieren Sie dies vor der Erstellung eines neuen Visual Studio Projektes.



Das Platform SDK dieses Projektes ist das Beckhoff HMI 600 SDK.

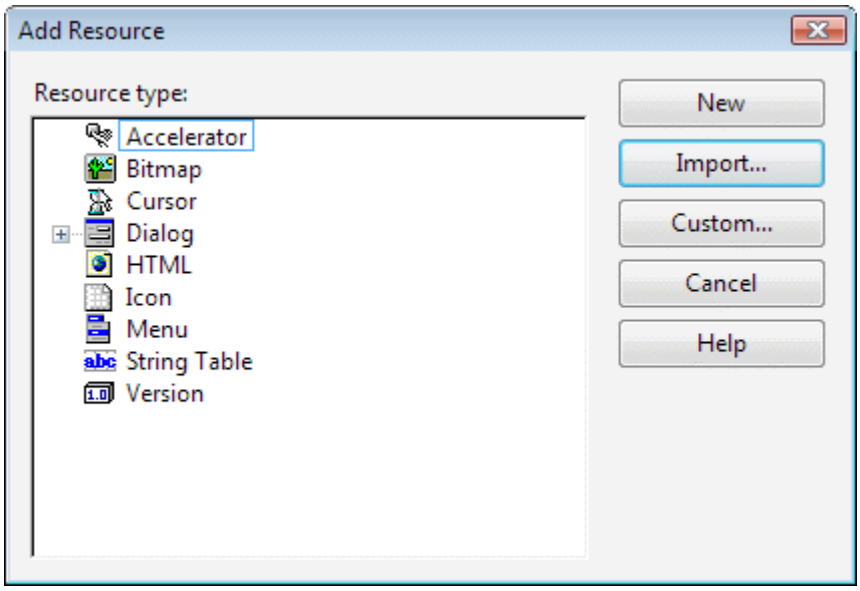


Als Servertyp wird Executable (EXE) ausgewählt.

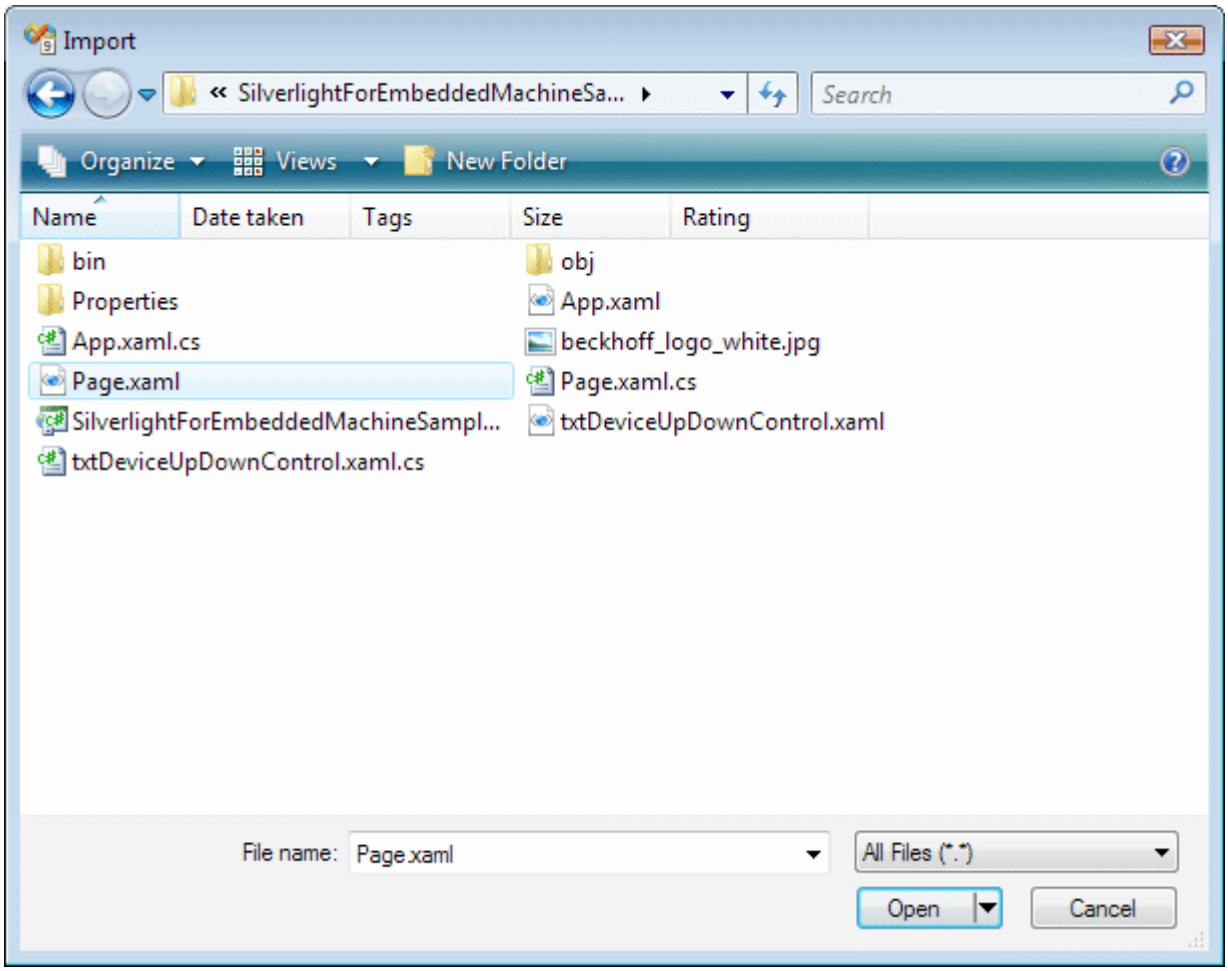


4. XAML-Datei als Resource einbinden

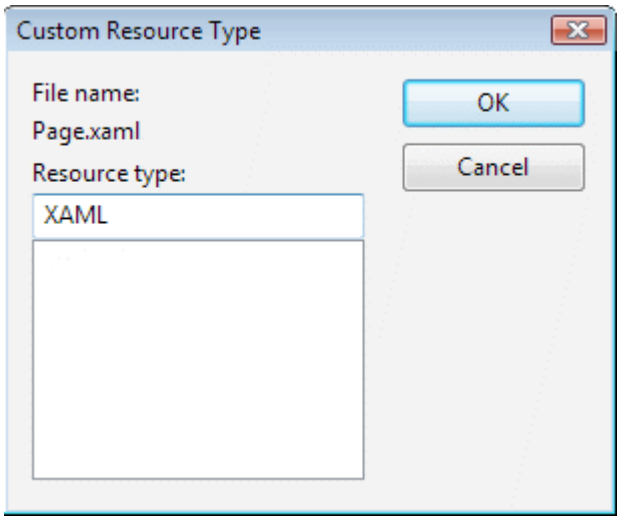
Die mit Expression Blend gestaltete Bedienoberfläche kann in das neue Projekt eingebunden werden. Öffnen Sie dazu die Resource-Datei (.rc). Mit einem Rechts-Klick auf die Resource im Resource View Tab und der Auswahl von *'Add -> Resource...'* öffnet sich ein Dialog über den die XAML-Datei eingebunden werden kann.



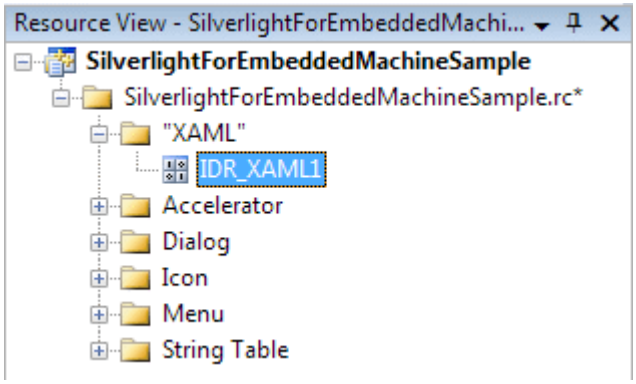
Mit Hilfe des Dialoges kann die XAML-Datei ins Projekt importiert werden.



XAML als Resourcetyp angeben.



Die Standard ID (IDR_XAML1) der Resource kann in diesem Beispiel beibehalten werden. In eigenen Projekten ist es jedoch sinnvoll diese umzubenennen.



5. AdsHelper Class erstellen
Der größte Teil der Ads-Kommunikation kann in einer separaten Klasse, hier AdsHelper genannt, ausgelagert werden.

AdsHelper.h
In den AdsHelper-Header werden die TcAds-Headers eingebunden. Dabei ist auf die richtige Pfadangabe zu achten. Zudem sind die Headers abhängig vom Prozessortyp. Im Beispielcode sind die X86-Headers angefügt.

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

Des Weiteren werden im Header die Deklarationen vorgenommen.

```
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
Die Header StdAfx.h und AdsHelper.h müssen in die AdsHelper.cpp eingebunden werden,

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

und anschließend werden die globalen Variablen definiert.

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

Die Methoden AdsGetVarHandle und AdsGetVarHandleEx dienen dazu, Handles zu SPS-Variablen zu erstellen

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

    unsigned long read = 0;
```

```
    long nErr =
    AdsSyncReadWriteReqEx(pServerAddrm, 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 =
    AdsSyncReadWriteReyEx2(port, pServerAddr, ADSIGRP_SYM_HNDBYNAME, 0x0,
    sizeof(long), pHandle, strlen(szVarname), (char*)szVarname, &read);

    return nErr;
}
```

Mit AdsFreeVarHandle und AdsFreeVarHandleEx werden Handles zu SPS-Variablen wieder freigegeben.

```
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_REALEASEHND, 0,
                 sizeof(pHandle), &pHandle);
}
```

Die folgenden beiden Methoden schreiben die SPS-Variable ".switch" und setzen dadurch die Geschwindigkeit auf langsam bzw. schnell.

```
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 setzenbool 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 setzenbool 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 der connect-Methode wird eine Verbindung zu den Variablen in der SPS erzeugt.

```
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 holenlong 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;
}
```

Beim Trennen der Ads-Verbindung müssen die Handles der SPS-Variablen freigegeben und der Port geschlossen werden.

```
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. Quellcode bearbeiten

In der SilverlightForEmbeddedMachineSample.cpp-Datei werden als erstes die Headers eingebunden.

```
#include"pwinuser.h"#include"xamlruntime.h"#include"xrdelegate.h"#include"xrptr.h"    #include"resou
rce.h"
```

Anschließend folgt die Variablendeklaration.

```
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);
```

Die Timer-Callback-Funktion dient zur Überprüfung der Ads-Verbindung und stellt bei Bedarf die Verbindung wieder her.

```
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;
}
```

Der folgende Ads-Event-Handler wird aufgerufen wenn sich eine SPS-Variable ändert, zu der eine Verknüpfung besteht.

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

Abhängig davon ob deviceUp, deviceDown oder engine auf TRUE gesetzt ist werden die entsprechenden Pfeile rot eingefärbt oder nicht.

Durch Verwendung von Timelines kann dieser Effekt noch verbessert werden.

```
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 gibt die Anzahl der Takte an. Der Wert wird über die Progressbar *prgSteps* ausgegeben. Hierzu müssen die Daten zunächst in ein Byte konvertiert werden, da die dazugehörige SPS Variable vom Typ Byte ist. Da der Progressbar nur Daten vom Typ float übergeben werden können erfolgt anschließend ein Konvertierung zum Typ float.

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

Wie auch schon bei Steps müssen bei *count* die Daten zunächst in ihren ursprünglichen Datentyp konvertiert werden, bevor sie in einen Text umgewandelt und dem Textblock *txtCount* übergeben werden können.

```
else if (User == count)
    {
    WCHAR text[6];
```

```
    wsprintf(text, L"%d", *(unsigned short*)handler->data);
    txtCount->SetText(text);
    }
```

Die Ausgabe des Geschwindigkeitstyps erfolgt über RadioButtons. Je nach Geschwindigkeit wird der entsprechende RadioButton markiert.

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

Der OnClick Event wird von den verschiedenen Instanzen angetriggert und über den Namen kann unterschieden werden, welche Instanz der Auslöser war.

```
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 (wcsmp(name, L"radSpeedSlow") == 0)
    {
        // Aufruf der Methode SpeedSlowEx um die Geschwindigkeit auf langsam zu setzen.
        adserror = SpeedSlowEx(port, &addr);
    }
    if (wscmp(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 der WinMain-Methode muss als erstes die XMAL-Runtime initialisiert werden. Ist XamlRuntimeInitialize erfolgreich, so wird die Silverlight for Windows Embedded Runtime in der Applikation gestartet.

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

Jede Silverlight for Windows Embedded Applikation hat ein einzelnes "Applikations"-Objekt über welches der Zugriff auf globale Eigenschaften möglich ist.
Um auf dieses Objekt zuzugreifen wird die GetXRApplicationInstance API verwendet.

```
    HRESULT retcode;

    // Load an dinit XAML resource
    IXRApplicationPtr app;

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

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

Nach der Initializierung des Applikation Objekts kann das Hauptfenster erstellt und Silverlight for Windows Embedded die Verwaltung des Objektes übergeben werden.

```
    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;
```

Das Objekt innerhalb einer Silverlight for Windows Embedded Applikation ist in Objektbäumen organisiert.
Um auf dieses Objekt zuzugreifen wird ein Pointer zum Wurzelelement benötigt.

```
    IXRFrameworkElementPtr root;

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

Erzeugen von Instanzen der Oberflächenelemente und der 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;
```

Die Gruppe "RadioButtonGroup" erstellen und die beiden RadioButtons dieser Gruppe zuweisen.

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

Zum Verbinden des EventHandlers mit den Buttons wird ein weiterleitendes Objekt benötigt.

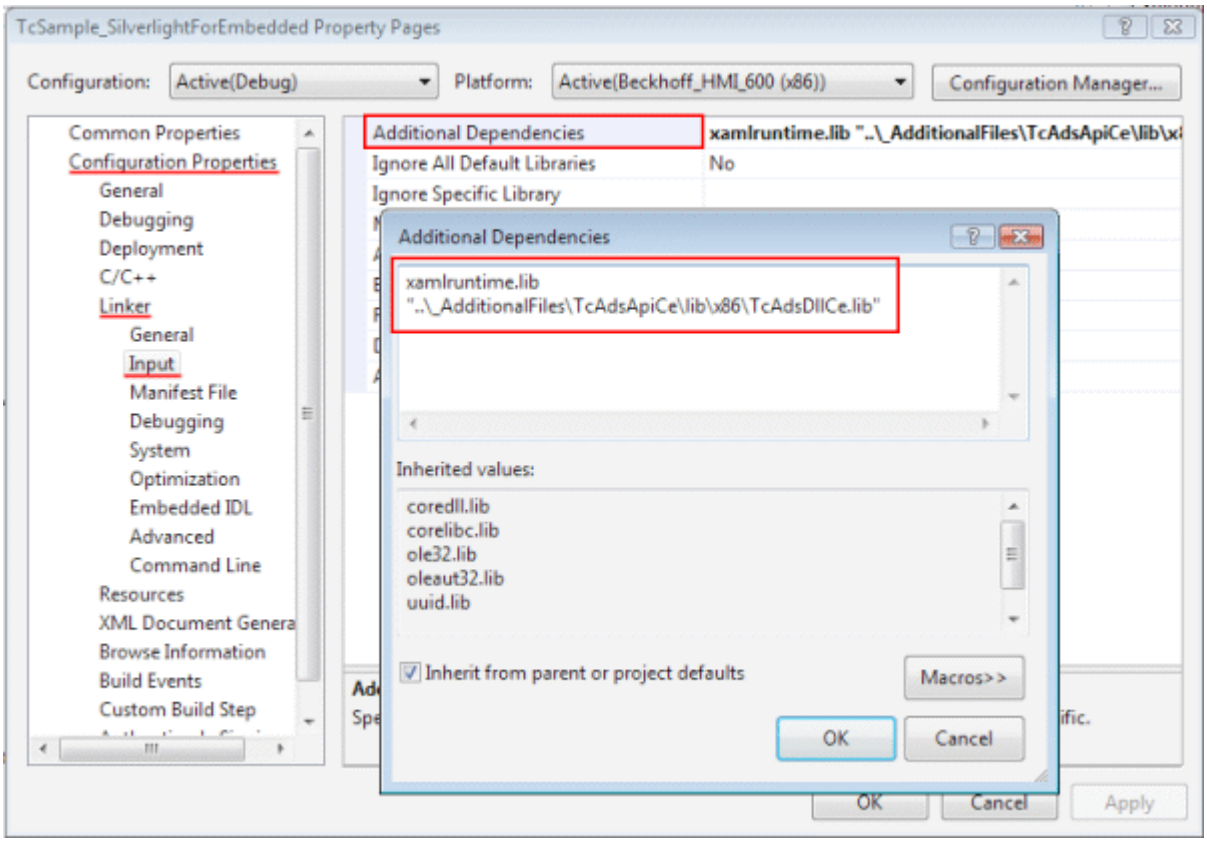
```
    BtnEventHandler handler;

    // 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;
```

Einbinden der Ads-Komponenten.

```
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. Eigenschaften

In den Projekteigenschaften muss eine Verbindung zur *xamlruntime.lib* und *TcAdsDllCe.lib* hergestellt werden.



Download Silverlight for Windows Embedded Beispiel

https://infosys.beckhoff.com/content/1031/tcsample_expression/Resources/12493869707.zip

Sample für ARM-Geräte

Im Sample wird die X86 Version der TcAdsDllCe.lib verwendet. Um das Sample für ARM-Geräte bauen zu können, muss diese Bibliothek vorher gegen die entsprechende ARM-Version ausgetauscht werden.

3.2 Beispiel Machine mit Microsoft Silverlight und JavaScript

Microsoft Silverlight ist eine Web-Präsentationstechnik, die durch ein entsprechendes Plugin von allen gängigen Browsern (Internet Explorer 6/7, Mozilla Firefox, Apple Safari und Opera) dargestellt werden kann.

Zielplattformen

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

Implementierung

- JavaScript

Erforderliche Software:

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

Welche Runtime Sie benötigen hängt davon ab, ob Sie eine Silverlight 1.0 oder 1.1 Applikation in Ihrem Browser darstellen wollen.

- **Developer Tools:**
  - Microsoft Visual Studio 2008 Beta 2
  - Microsoft Silverlight Tools Alpha Refresh for Visual Studio (July 2007)
- Oder
  - Microsoft Visual Studio 2005
  - Microsoft Silverlight 1.0 Software Development Kit

Für dieses Beispiel wurde Microsoft Visual Studio 2005 verwendet.

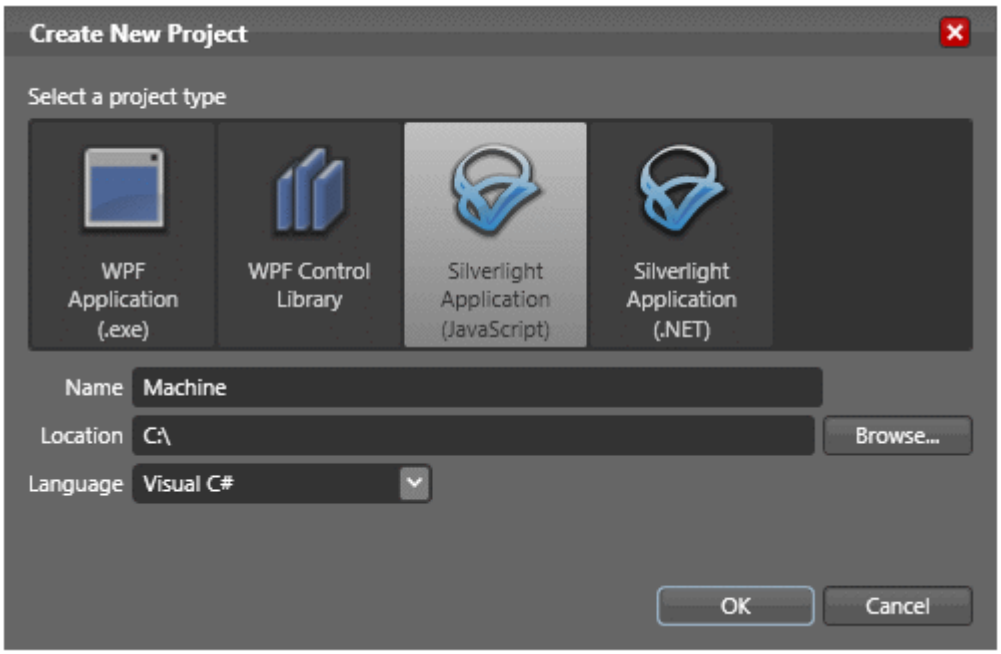
- **Designer Tools:**
  - Expression Blend 2 August Preview
- **Sonstige:**
  - TwinCAT 2.10
  - Browser (z.B. Internet Explorer 7 oder Mozilla Firefox)
  - Microsoft .NET Framework Version 3.0

Die ersten Schritte...

Schritt für Schritt lernen Sie die Entwicklung einer Silverlight-Applikation und das Einbinden des TwinCAT ADS Web Service anhand eines Beispiels kennen.

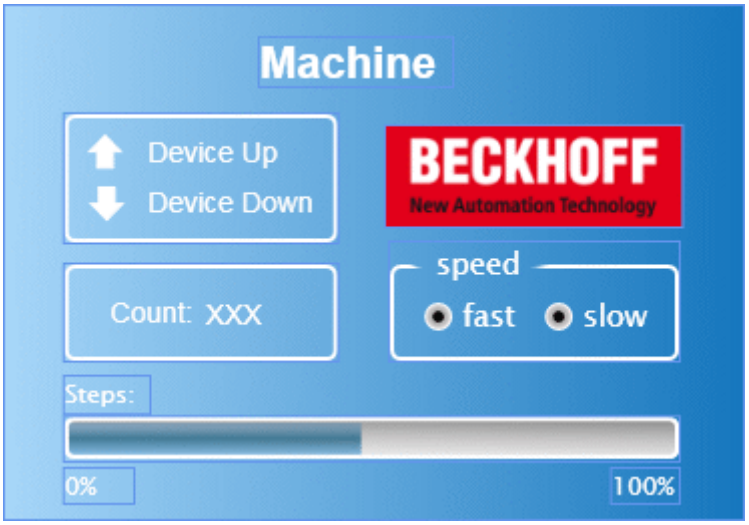
1. Neues Projekt erstellen:

Starten Sie Microsoft Expression Blend 2 und erstellen Sie eine neue Oberfläche über 'Menü → File → new Project...' . Das Dialogfeld 'Create New Projekt' öffnet sich und es kann nun der Typ, der Name, der Speicherort und die Programmiersprache ausgewählt werden. In diesem Beispiel wählen Sie den Typ 'Silverlight Application (JavaScript)' und den Namen 'Machine'.



2. Bedienoberfläche erstellen:

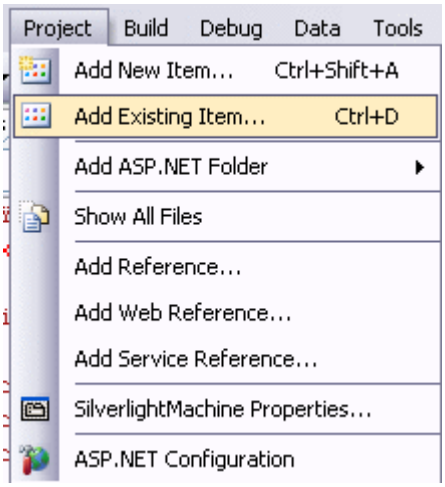
Für die Erstellung der Bedienoberfläche stehen nur wenige Controls zur Verfügung. Einen *RadioButton* können Sie mit einer *Textbox* und zwei *Ellipsen* erstellen, die in einem *Canvas* gepackt werden. Die *Progressbar* kann man mit drei *Rectangles* erstellen, die man auch wieder in ein separates *Canvas* packt. Die Einstellungen der Oberfläche werden in der Datei *Page.xaml* abgelegt.
Beachten Sie, dass Sie bei keinem Objekt die Größe auf 'Auto' setzen. Dies kann sonst später zu Fehlern führen.



Im oberen linken Bereich sehen Sie die beiden Ausgänge, die auch auf den Busklemmen ausgegeben werden. Unten links ist die Variable abgebildet, welche die Werkstücke zählt. Rechts können Sie mit dem Feld *Speed* die Taktgeschwindigkeit des Motors verändern. Die Anzeige *Steps* entspricht der Anzahl der Takte, die auf den Ausgang 1 ausgegeben werden.

3. Add XMLHTTP.JS

Im Visual Studio über 'Projekt → Add Existing Item...' fügen Sie die Datei 'XMLHTTP.JS' ein. Diese Datei enthält allgemeine Methoden zum Lesen und Schreiben von SPS-Variablen, sowie zum Konvertieren von Datentypen.



4. Quelltext bearbeiten

Öffnen Sie die Datei Default.html in Visual Studio und binden Sie im Kopf die Datei 'XMLHTTP.JS' ein.

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

Fügen Sie in der HTML Seite im HEAD-Bereich einen JavaScript-Bereich ein. Der folgende Quelltext muss dort eingefügt werden:

Zunächst müssen die wichtigsten Variablen deklariert werden.

```
<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;

...
```

Die URL des TcAdsWebService sowie die NetID und den Port müssen Sie entsprechend anpassen.

Auf die Objekte der Bedienoberfläche kann nicht ohne weiteres zugegriffen werden. Damit dies funktioniert werden die Objekte nach dem Start der Applikation den Variablen zugewiesen, die oben bereits deklariert wurden.

```
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");
}
```

Die Load Methode beinhaltet zwar das Zuweisen der Variablen, jedoch wird diese bisher nie aufgerufen. Damit dies geschieht muss in Expression Blend 2 der XAML-Code der Oberfläche geändert werden. Dazu muss das oberste *Canvas* um *Loaded="Load"* ergänzt werden.

```
<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);
}
```

SPS-Variable lesen

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

- **netId:** String, der die AMS-Net-Id angibt, auf der die SPS zu finden ist
- **nPort:** Port-Nummerdes Laufzeitsystems
- **indexGroup:** IndexGroup der SPS-Variable
- **indexOffset:** Erstes Byte, in das geschrieben werden soll
- **ncbLen:** Anzahl der Bytes, die geschrieben werden sollen

```
function init()
{
     b64s = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=";
     success = 0;
     errors = 0;
     loop(refresh); //send request every x seconds
}
```

Ergänzen Sie *<body>* um *onload="init()"*, damit die Methode beim Laden ausgeführt wird.

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

Die nächste Funktion sorgt dafür, dass die Werte ausgelesen und ausgeben werden. Wichtig ist hier, dass die Adresse der Variable von Machine.pro zum Auslesen richtig angegeben wird.

```
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 den letzten beiden Methoden wird die SPS-Variable, über die die Geschwindigkeit der Maschine gesteuert wird, auf null bzw. eins gesetzt.

```
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');
}
```

SPS Variablen schreiben

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

- **netId:** String, der die AMS-Net-Id angibt, auf der die SPS zu finden ist
- **nPort:** Port-Nummerdes Laufzeitsystems
- **indexGroup:** IndexGroup der SPS-Variable
- **indexOffset:** Erstes Byte, in das geschrieben werden soll
- **ncbLen:** Anzahl der Bytes, die geschrieben werden sollen
- **pwrData:** Array, dass die zu schreibenden Daten enthält
- **type:** "bool", "int" oder "string"

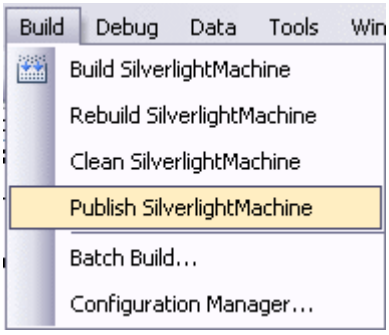
Wie auch bei der 'Load' Funktion muss auch hier nach Expression Blend 2 gewechselt werden, um in den entsprechenden Zeilen die beiden Methoden zum 'Klick-Event' ihrer beiden Buttons zu machen.

```
<Canvas x:Name="canvasFast" MouseLeftButtonDown="Fast_MouseLeftButtonDown" ...
```

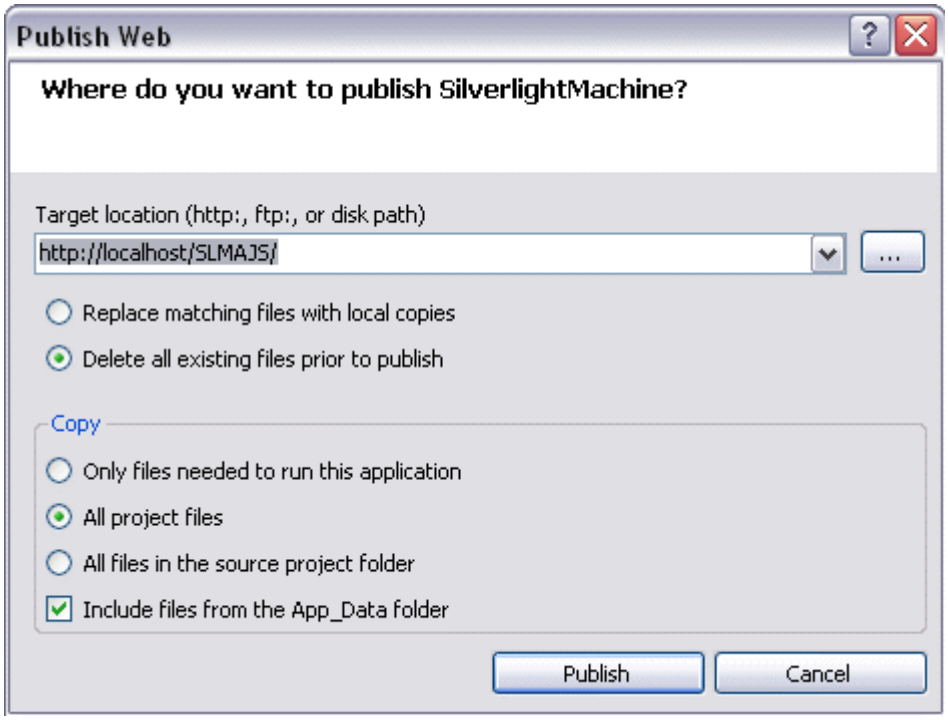
```
<Canvas x:Name="canvasSlow" MouseLeftButtonDown="Slow_MouseLeftButtonDown" ...
```

5. Testen:

Debuggen Sie zunächst Ihre Applikation. Sie werden feststellen, dass diese nicht so funktioniert wie Sie es erwarten. Anschließend gehen Sie über 'Build → Publish'.



Wählen Sie im Dialogfenster die 'Target location', sowie unter 'Copy' 'All project files' aus.



Wenn unten links in der Statusbar 'Publish succeded' steht, können Sie Ihre Anwendung in einem Browser ausführen und testen.

Download:

https://infosys.beckhoff.com/content/1031/tcsample_expression/Resources/12493871115.zip

4 Beispiele WPF

Windows Presentation Foundation

- Zielplattformen: XP, XPE, WES, Vista, Win 7
- Implementierung: Visual C#, Visual Basic

Beispiele WPF

Beschreibung	Beispiel
Beispiel 1: Machine WPF C# Sample [▶ 29]	https://infosys.beckhoff.com/content/1031/tcsample_expression/Resources/12493866891.zip
Beispiel 2: Machine WPF Visual Basic Sample [▶ 36]	https://infosys.beckhoff.com/content/1031/tcsample_expression/Resources/12493868299.zip

4.1 Beispiel Machine mit Microsoft Expression Blend (C#)

Microsoft Expression Blend ist ein Programm zum Erstellen von Programmoberflächen für C# und Visual Basic. In diesem Beispiel wird eine, mit dem Programm erstellte, Oberfläche mit dem Beispiel Machine verbunden und das Ganze anschließend in den Vista Media Center eingebunden. Dabei wurde die Programmiersprache C# verwendet.

Zielplattform

- Windows Vista

Implementierung

- Visual C#

Erforderliche Software

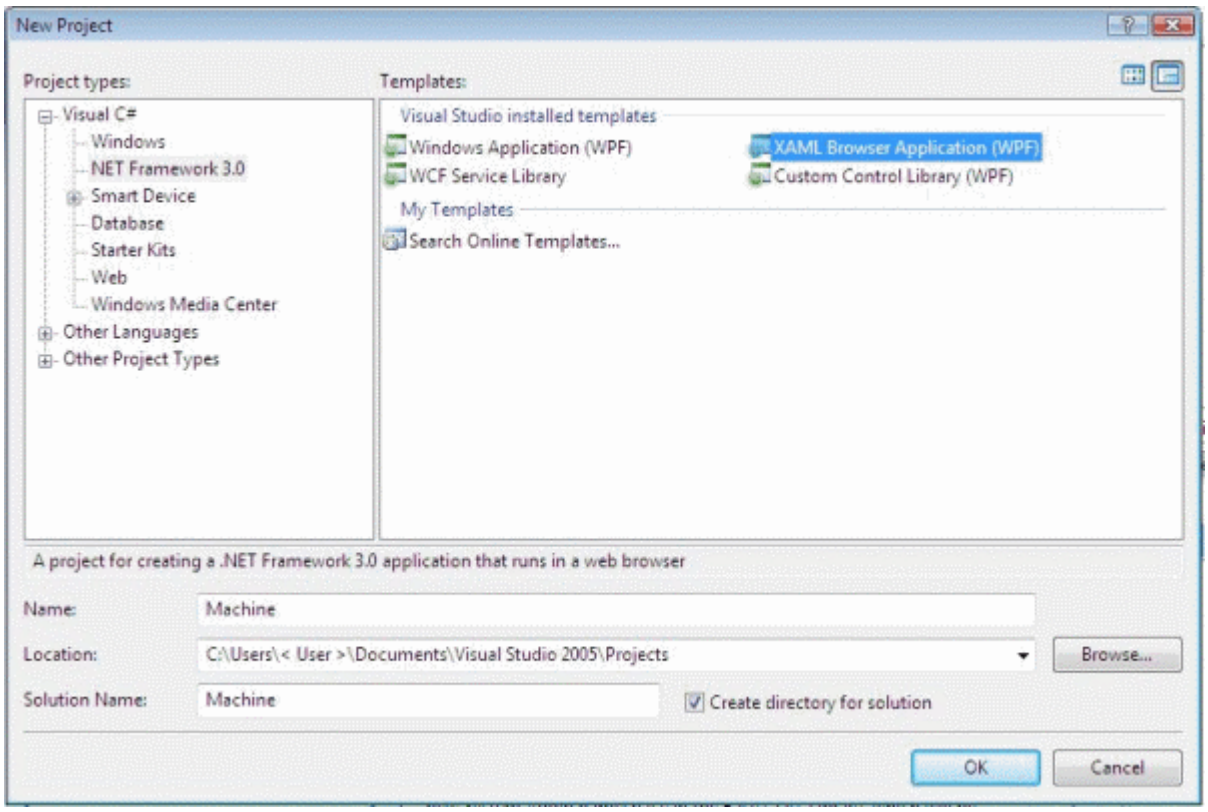
- Microsoft .NET Framework Version 3.0
- Microsoft Expression Blend
- Microsoft Visual Studio 2005
- Microsoft Windows Vista Media Center
- Microsoft Windows SDK für .Net Framework 3.0
- Microsoft Visual Studio 2005 extensions für .Net Framework 3.0 (November 2006 CTP)
- TwinCAT 2.10
- Notepad oder einen anderen Texteditor

Die ersten Schritte ...

Schritt für Schritt lernen Sie die Entwicklung eines Programms mit Microsoft Visual Studio und Microsoft Expression Blend und das Einbinden der TwinCAT ADS .NET Komponente anhand eines Beispiels kennen und binden dieses dann in den Vista Media Center ein.

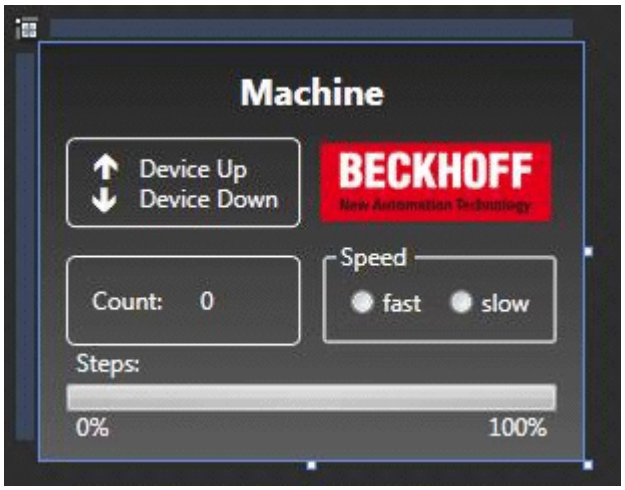
1. Neues Projekt erstellen:

Starten Sie Microsoft Visual Studio und erstellen Sie ein neue XAML Browser Applikation. Dazu gehen Sie über das Menü unter 'File -> New -> Project...' . Das Dialogfeld 'New Project' öffnet sich. Zuerst wählen Sie den Projekt Typ aus: 'Project types -> Visual C# -> Net Framework 3.0'. Rechts erscheinen dann die Templates des Projekt Typs. Dort wählen Sie 'XAML Browser Application'. Sie geben Ihrem Projekt jetzt noch einen Namen, in diesem Fall ist das Machine und legen die Location fest.



2. Bedienungsoberfläche erstellen

Sie wechseln jetzt zu Microsoft Expression Blend, wo Sie Ihr eben erstelltes Projekt öffnen, um dort die Bedienoberfläche zu erstellen.

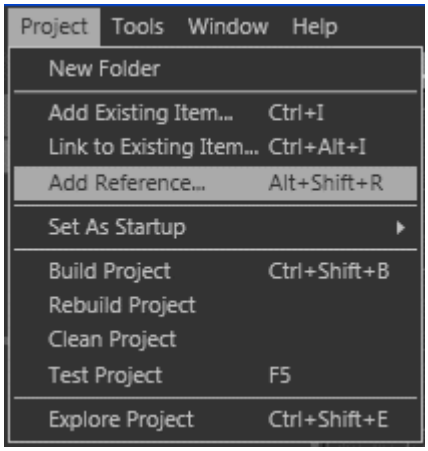


Im oberen linken Bereich sehen Sie die beiden Ausgänge, die auch auf den Busklemmen ausgegeben werden. Unten links ist die Variable abgebildet, welche die Werkstücke zählt. Rechts können Sie mit dem Feld 'Speed' die Taktgeschwindigkeit des Motors verändern. Die Anzeige 'Steps' entspricht der Anzahl der Takte, die auf den Ausgang 1 ausgegeben werden.

Wollen Sie, dass die Bedienoberfläche ständig ihre Größe anpasst, so kopieren Sie den obersten Grid und fügen dann anstelle des Grids eine Viewbox ein. In diese Viewbox fügen Sie nun den Grid ein. Jetzt müssen Sie nur noch die Größe der Seite, der Viewbox und des Grids auf 'Auto' setzen. Hierbei kann es zu einer Verschiebung von Elementen kommen. Diese müssen Sie dann erneut positionieren. Achten Sie dabei darauf, dass Sie die Größe der Seite, der Viewbox und des Grids nicht aus Versehen wieder festsetzen.

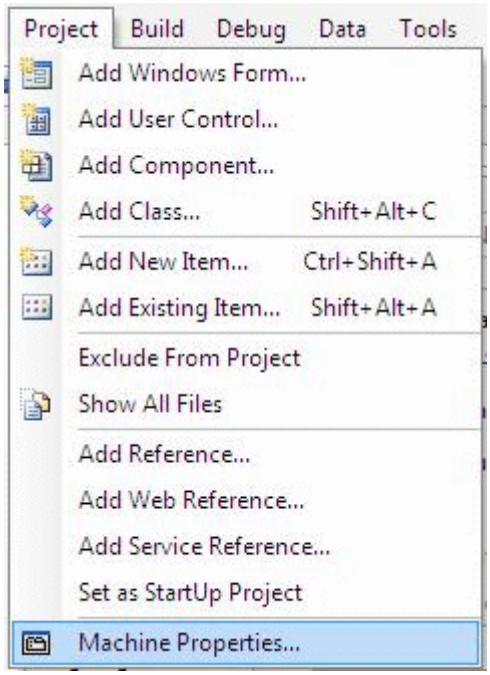
3. Referenz hinzufügen

Nach dem Erstellen der Oberfläche muss zuerst eine Referenz namens 'TwinCAT.Ads.dll' hinzugefügt werden. Dies kann sowohl in Visual Studio, als auch in Expression Blend erfolgen. In beiden Fällen geht man über das Menü unter 'Project --> Add Reference'.

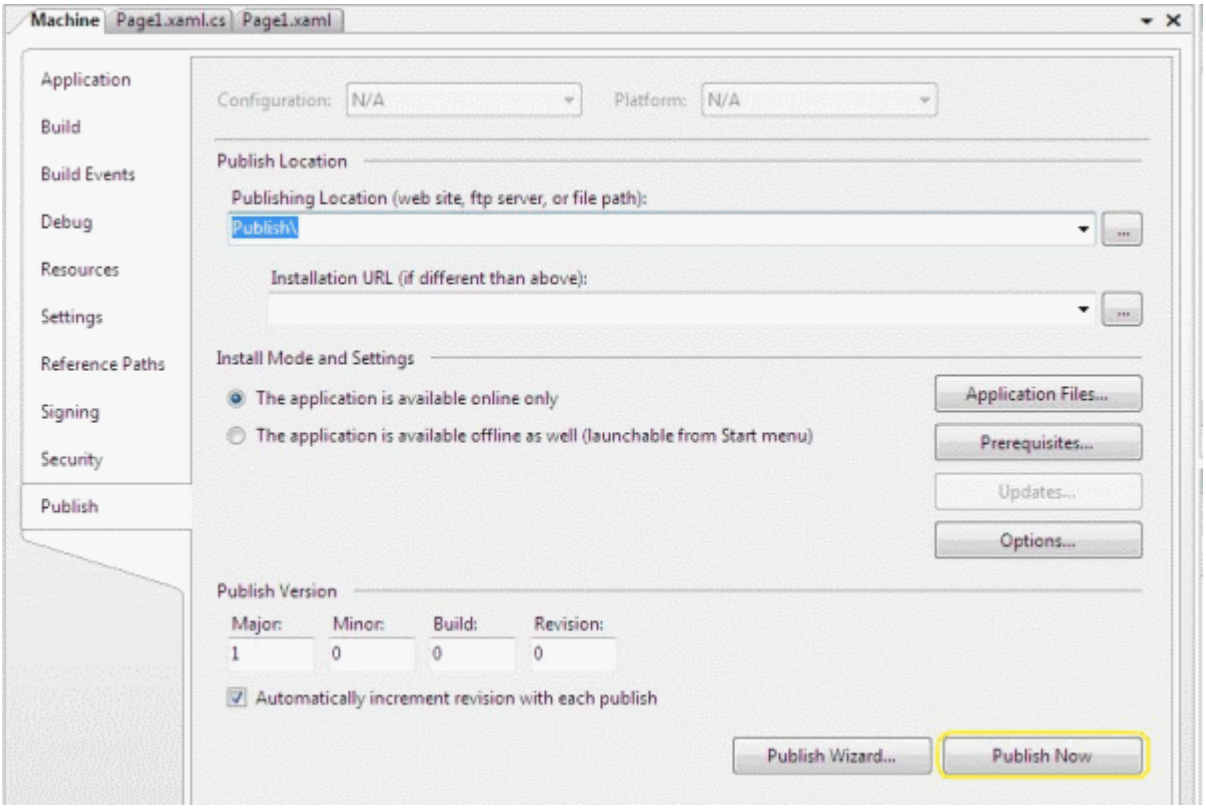


4. Sicherheitsfreigabe

Gehen Sie im Menü unter 'Project -> <Name ihres Projektes> Properties....' .



Es öffnet sich nun ein Tab, in dem Sie die Projekteigenschaften festlegen können. Gehen Sie dort auf 'Security' und wählen Sie 'this is a full trust application' aus.



5. Quelltext bearbeiten

Nun kann mit dem Erstellen des Quelltextes in C# begonnen werden.
In die oberste Zeile des Quelltextes werden die benötigten Namespaces 'System.IO' und 'TwinCAT.Ads' eingefügt.

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

Danach folgen die Deklarationen.

```
privateTcAdsClient tcClient;
privateAdsStream dataStream;
privateBinaryReader binReader;
private int hEngine;
private int hDeviceUp;
private int hDeviceDown;
private int hSteps;
private int hCount;
private int hSwitchNotify;
private int hSwitchWrite;
```

Die erste Methode, ist die 'Load' Methode. In ihr werden Instanzen verschiedener Klassen erzeugt und eine Verbindung zum Port 801 hergestellt.

```
//------------------------------------------------------// 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 cl
ass
     dataStream = new AdsStream(7);

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

     // Eine neue Instanz der Klasse TcAdsClient erzeugen// Create an new instance of the TcAdsClien
t 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");
    }

    //...
```

Dann werden in der Methode 'Load' die Variablen noch verbunden und mit einer Methode (die noch geschrieben werden muss) verknüpft, die bei einer Änderung einer Variable aufgerufen wird.

```
try
    {
    // Initialisieren der Überwachung der SPS-
Variablen// Initalizing 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

SPS Variablen verbinden:
Zum Verbinden der Variablen wurde die Methode AddDeviceNotification verwendet.

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

- **variableName:** Name der SPS Variablen.
- **dataStream:** Der Datenstrom, der die Daten empfängt.
- **offset:** Abstand im Datenstrom.
- **length:** Länge im Datenstrom.
- **transMode:** Das Ereignis, wenn sich die Variable ändert.
- **cycletime:** Die Zeit (in ms) nach der der SPS-Server überprüft, ob sich die Variable geändert hat.
- **maxDelay:** Die Zeit (in ms) nach der das Ereignis spätestens beendet ist.
- **userData:** Das Objekt, das zum Ablegen bestimmter Daten verwendet werden kann.

Zum Verbinden der Variable 'hSwitchWrite' wurde die Methode CreateVariableHandle verwendet.

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

- **variableName:** Name der SPS-Variablen.

7. Methode schreiben:

Oben wurde bereits auf eine Methode verwiesen, die noch gar nicht existiert. Daher wird diese Methode, die 'tcClient_OnNotification' genannt wurde, als nächstes geschrieben. Diese Methode wird aufgerufen, wenn sich eine der SPS-Variable geändert hat.

```
//-----------------------------------------------// 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);
    }
}
```

Es fehlen noch zwei Methoden, mit denen die Geschwindigkeit der Maschine eingestellt werden kann. In Ihnen wird ein virtueller Schalter umgelegt, hier wird ein Wert in die SPS-Variable switch geschrieben.

```
//-------------------------------------------------------// 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. Notifications und Handles löschen:

In dem Close-Ereignis des Fensters werden die Verbindungen wieder mit der Methode DeleteDeviceNotification() freigegeben.

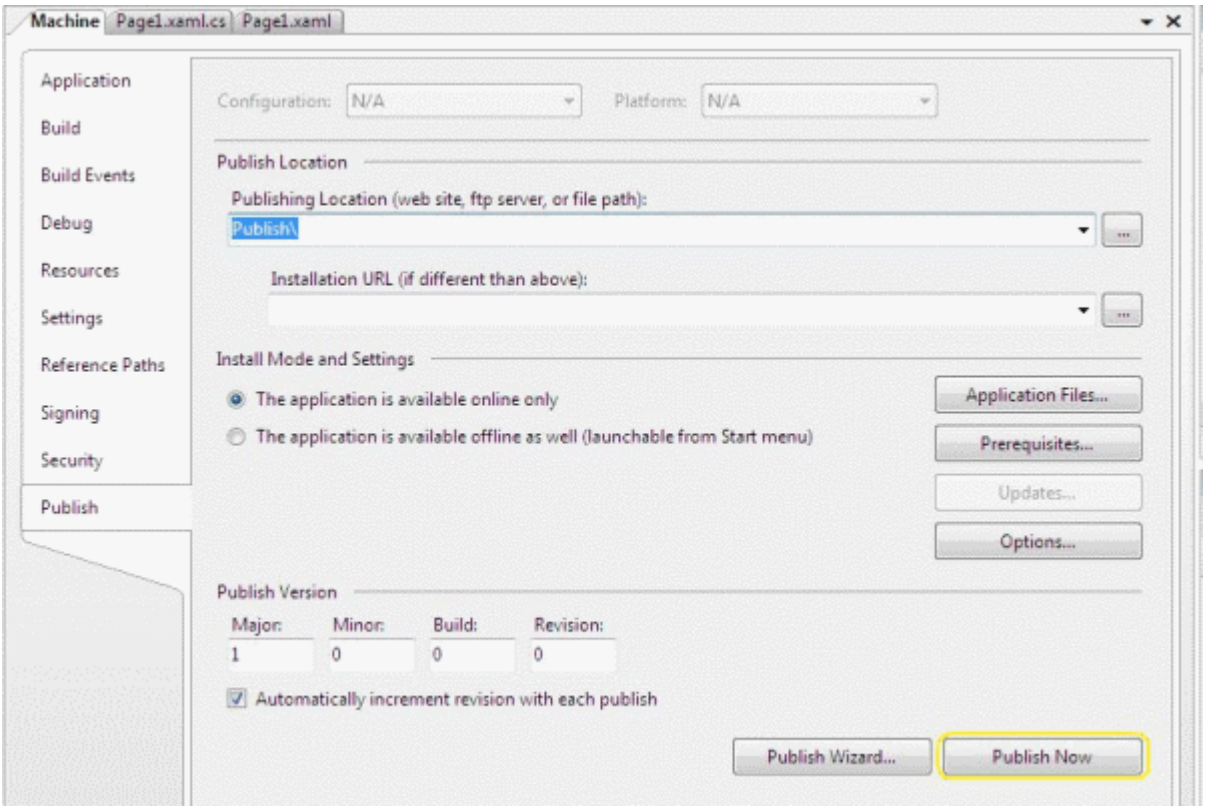
```
//-------------------------------------------------------// wird beim Beenden des Programms aufgerufe
n// is activated when ending the program//-----------------------------------------------------
private void Close(object sender, EventArgs e)
{
    try
    {
    // Löschen der Notifications und 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();
}
```

Zu guter Letzt muss noch dafür gesorgt werden, dass die Methoden auch beim richtigen Ereignis aufgerufen werden. Dazu in Expression Blend gehen, Page auswählen, bei Properties auf Events wechseln und bei 'Loaded' 'Load' reinschreiben und bei 'Unloaded' 'Close'.
Das Gleiche muss auch noch mit den beiden RadioButtons gemacht werden, nur das hierbei das Event 'Click' ausgewählt wird, sowie die Methode 'optSpeedFast_Click' bzw. 'optSpeedSlow_Click'.
Das SPS Maschinenprogramm Machine_Final.pro muss auf dem Laufzeitsystem 1 laufen und das Programm kann im Internet Explorer 7 getestet werden.

9. Einbinden in den Vista Media Center

Habe Sie ihr Projekt hinreichend getestet und keinen Fehler festgestellt, dann können Sie es nun in den Media Center einbinden.
Rufen Sie wieder in Visual Studio die Projekteigenschaften auf, jedoch gehen Sie dann auf 'Publish'. Dort klicken Sie dann auf 'Publish Now'. Es wird nun die xbap-Datei erstellt, die Sie später im Media Center aufrufen. Diesen Schritt müssen Sie immer machen, wenn Sie ihr Programm verändert haben und die

Änderung auch in den Media Center übernommen werden soll.



Gehen Sie jetzt in einen Texteditor, z. B. Notepad und geben Sie folgendes ein:

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

Speichern Sie es unter: 'C:\Users\<User>\AppData\Roaming\Media Center Programs\Machine.mcl'. Wenn Sie nun Ihren Media Center starten finden Sie ihr Programm unter 'Online Media -> program library -> programs by name -> Machine'.
Dies ist die einfachste Form der Einbindung in den Windows Vista Media Center. Weitere Informationen zur Einbindung in den Media Center finden Sie hier.

10. Download Expression Blend Beispiel:

https://infosys.beckhoff.com/content/1031/tcsample_expression/Resources/12493866891.zip

4.2 Beispiel Machine mit Microsoft Expression Blend (VB)

Microsoft Expression Blend ist ein Programm zum Erstellen von Programmoberflächen für C# und Visual Basic. In diesem Beispiel wird eine, mit dem Programm erstellte Oberfläche mit dem Beispiel Machine verbunden und anschließend in den Vista Media Center eingebunden. Dabei wird die Programmiersprache Visual Basic verwendet.

Zielplattform

- Windows Vista

Implementierung

- Visual Basic

Erforderliche Software

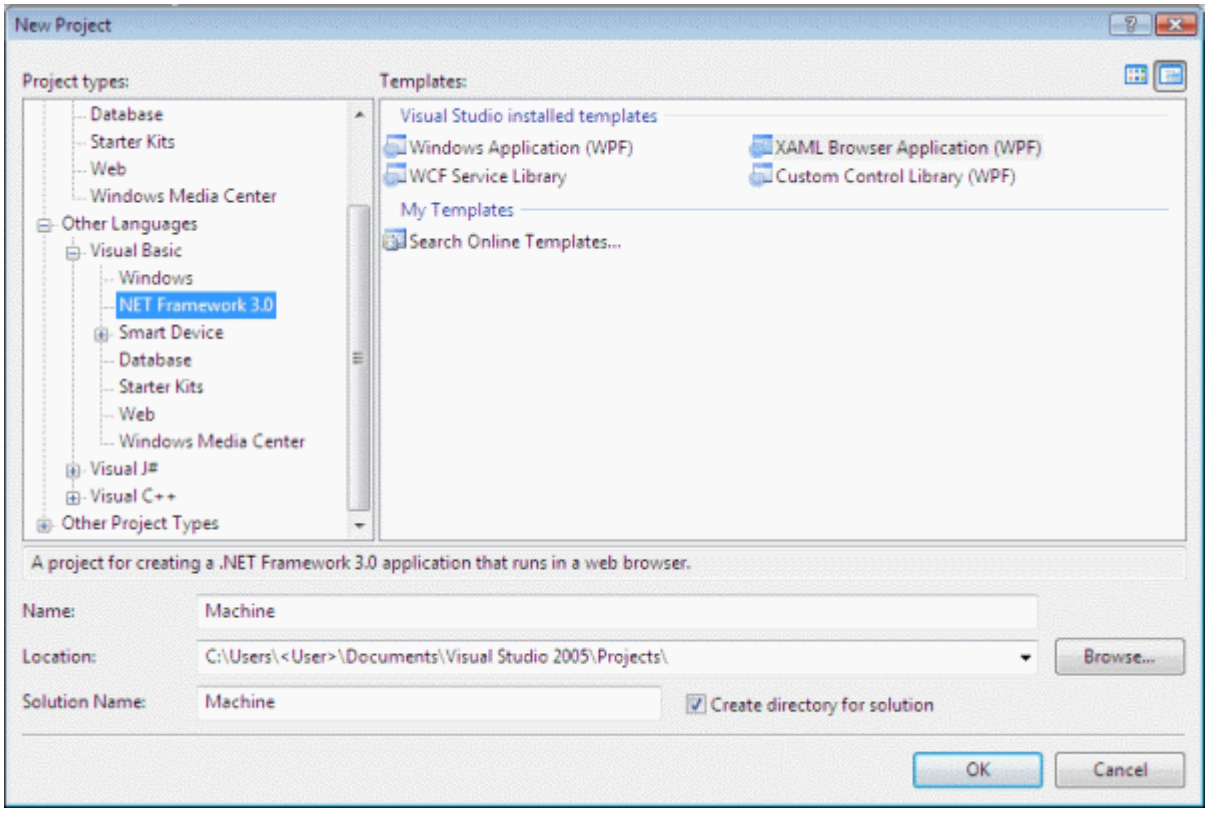
- Microsoft .NET Framework Version 3.0
- Microsoft Expression Blend, mehr dazu hier
- Microsoft Visual Studio 2005
- Microsoft Windows Vista Media Center
- Microsoft Windows SDK für .Net Framework 3.0
- Microsoft Visual Studio 2005 extensions für .Net Framework 3.0 (November 2006 CTP)
- TwinCAT 2.10
- Notepad oder einen anderen Texteditor

Die ersten Schritte ...

Schritt für Schritt lernen Sie die Entwicklung eines Programms mit Microsoft Visual Studio und Microsoft Expression Blend und das Einbinden der TwinCAT ADS .NET Komponente anhand eines Beispiels kennen und binden dieses dann in den Vista Media Center ein.

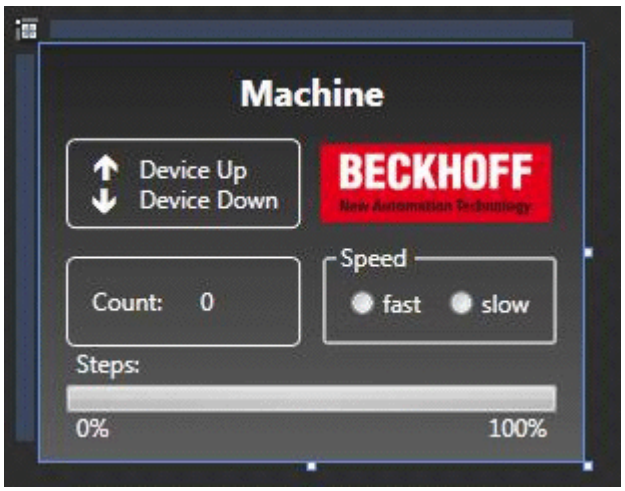
1. Neues Projekt erstellen:

Starten Sie Microsoft Visual Studio und erstellen Sie ein neue XAML Browser Applikation. Dazu gehen Sie über das Menü unter 'File -> New -> Project...' . Das Dialogfeld 'New Project' öffnet sich. Zuerst wählen Sie den Projekt Typ aus: 'Project types -> Visual Basic -> Net Framework 3.0'. Rechts erscheinen dann die Templates des Projekt Typs. Dort wählen Sie 'XAML Browser Application'. Sie geben Ihrem Projekt jetzt noch einen Namen, in diesem Fall ist 'Machine' und legen die Location fest.



2. Bedienungsoberfläche erstellen

Wechseln Sie jetzt zu Microsoft Expression Blend, wo Sie Ihr eben erstelltes Projekt öffnen, um dort die Bedienoberfläche zu erstellen.

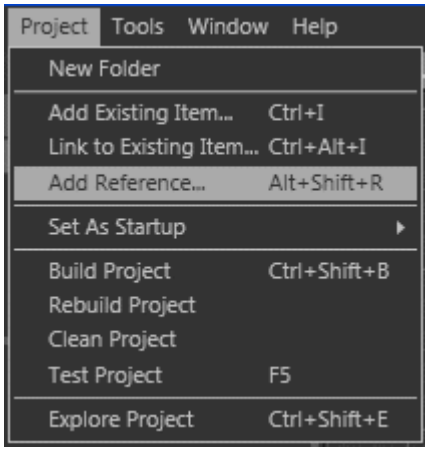


Im oberen linken Bereich sehen Sie die beiden Ausgänge, die auch auf den Busklemmen ausgegeben werden. Unten links ist die Variable abgebildet, welche die Werkstücke zählt. Rechts können Sie mit dem Feld 'Speed' die Taktgeschwindigkeit des Motors verändern. Die Anzeige 'Steps' entspricht der Anzahl der Takte, die auf den Ausgang 1 ausgegeben werden.

Wollen Sie, dass die Bedienoberfläche ständig ihre Größe anpassen, so kopieren Sie den obersten Grid und fügen dann anstelle des Grids eine Viewbox ein. In diese Viewbox fügen Sie nun den Grid ein. Jetzt müssen Sie nur noch die Größe der Seite, der Viewbox und des Grids auf 'Auto' setzen. Hierbei kann es zu einer Verschiebung von Elementen kommen. Diese müssen Sie dann erneut positionieren. Achten Sie dabei darauf, dass Sie die Größe der Seite, der Viewbox und des Grids nicht wieder festsetzen.

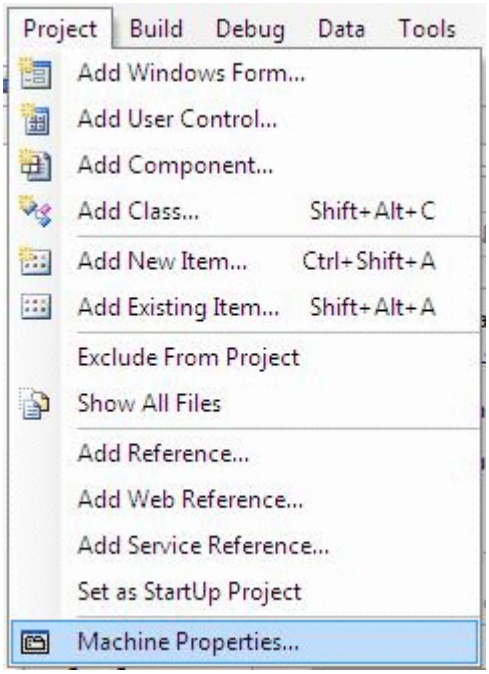
3. Referenz hinzufügen

Nach dem Erstellen der Oberfläche muss zuerst eine Referenz namens 'TwinCAT.Ads.dll' hinzugefügt werden. Dies kann sowohl in Visual Studio, als auch in Expression Blend erfolgen. In beiden Fällen geht man über das Menü unter 'Project --> Add Reference'.

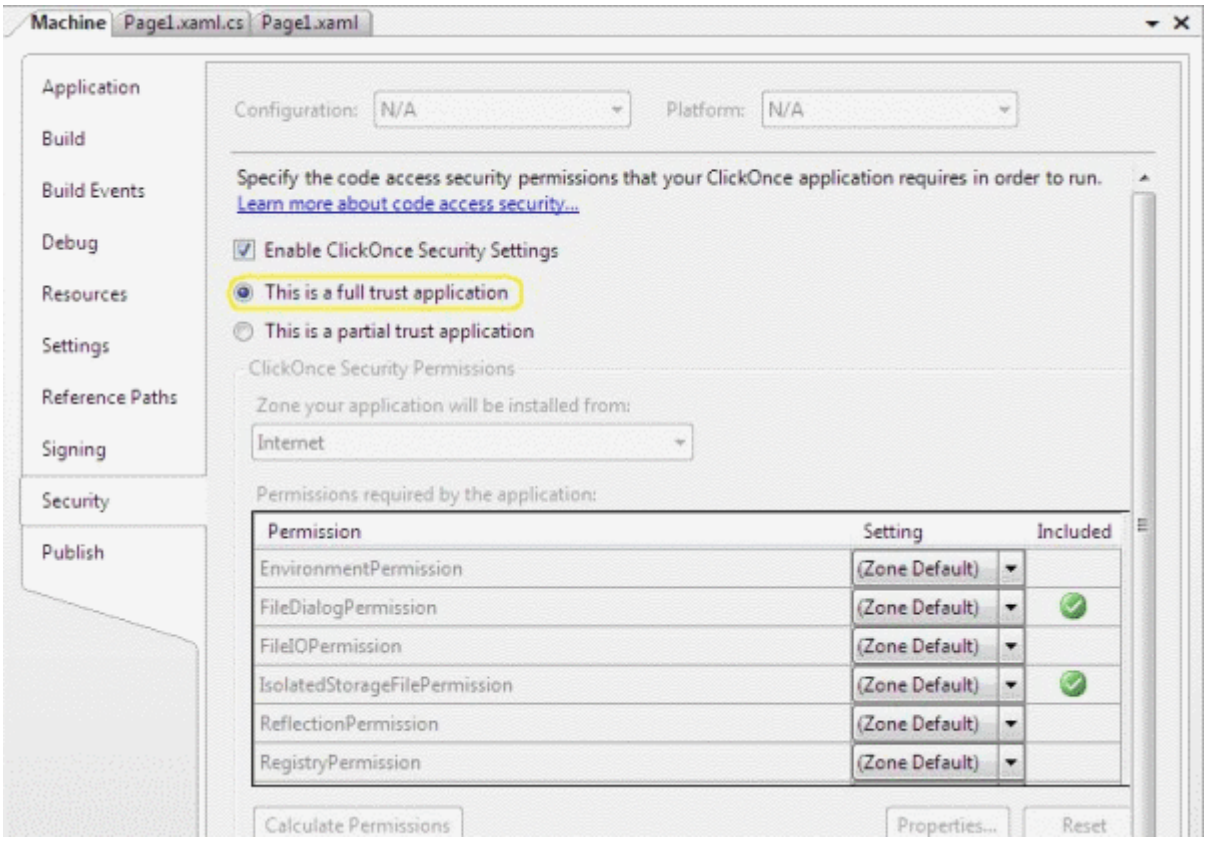


4. Sicherheitsfreigabe

Gehen Sie im Menü unter 'Project -> <Name ihres Projektes> Properties....' .



Es öffnet sich nun ein Tab, in dem Sie die Projekteigenschaften festlegen können. Gehen Sie dort auf 'Security' und wählen Sie 'this is a full trust application' aus.



5. Quelltext bearbeiten

Nun kann mit dem Erstellen des Quelltextes in C# begonnen werden.
In die oberste Zeile des Quelltextes werden die benötigten Namespaces 'System.IO' und 'TwinCAT.Ads' eingefügt.

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

Danach folgen die Deklarationen.

```
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
```

Die erste Methode, ist die 'Load' Methode. In ihr werden Instanzen verschiedener Klassen erzeugt und eine Verbindung zum Port 801 hergestellt.

```
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
...
```

Dann werden in der Methode 'Load' die Variablen noch verbunden und mit einer Methode (die noch geschrieben werden muss) verknüpft, die bei einer Änderung einer Variable aufgerufen wird.

```
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.OnCha
nge, 10, 0, DBNull.Value)
        hDeviceDown = tcClient.AddDeviceNotification(".deviceDown", dataStream, 2, 1, AdsTransMode.O
nChange, 10, 0, DBNull.Value)
        hSteps = tcClient.AddDeviceNotification(".steps", dataStream, 0, 1, AdsTransMode.OnChange, 1
0, 0, DBNull.Value)
        hCount = tcClient.AddDeviceNotification(".count", dataStream, 4, 2, AdsTransMode.OnChange, 1
0, 0, DBNull.Value)
        hSwitchNotify = tcClient.AddDeviceNotification(".switch", dataStream, 6, 1, AdsTransMode.OnC
hange, 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.AdsNotificatio
n, AddressOf tcClient_OnNotification
    Catch
        MessageBox.Show("Error when connecting")
    End Try
    End Sub
```

6. Definition

SPS Variablen verbinden:
Zum Verbinden der Variablen wurde die Methode AddDeviceNotification verwendet.

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

- **variableName:** Name der SPS Variablen.
- **dataStream:** Der Datenstrom, der die Daten empfängt.
- **offset:** Abstand im Datenstrom.
- **length:** Länge im Datenstrom.
- **transMode:** Das Ereignis, wenn sich die Variable ändert.
- **cycletime:** Die Zeit (in ms) nach der der SPS-Server überprüft, ob sich die Variable geändert hat.
- **maxDelay:** Die Zeit (in ms) nach der das Ereignis spätestens beendet ist.
- **userData:** Das Objekt, das zum Ablegen bestimmter Daten verwendet werden kann.

Zum Verbinden der Variablen 'hSwitchWrite' wurde die Methode CreateVariableHandle verwendet.

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

- **variableName:** Name der SPS-Variablen.

7. Methode schreiben:

Oben wurde bereits auf eine Methode verwiesen, die noch gar nicht existiert. Daher wird diese Methode, die 'tcClient_OnNotification' genannt wurde, als nächstes geschrieben. Diese Methode wird aufgerufen, wenn sich eine der SPS-Variable geändert hat.

```
'-----------------------------------------------' 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 entsprechened 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"-Werts
        ' 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
```

Es fehlen noch zwei Methoden, mit denen die Geschwindigkeit der Maschine eingestellt werden kann. In Ihnen wird ein virtueller Schalter umgelegt, hier wird ein Wert in die SPS-Variablen switch geschrieben.

```
'-----------------------------------------------------' 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) Ha
ndles optSpeedSlow.Click
    Try
        tcClient.WriteAny(hSwitchWrite, False)
    Catch
        MessageBox.Show("Error")
    End Try
End Sub
```

8. Notifications und Handles löschen:

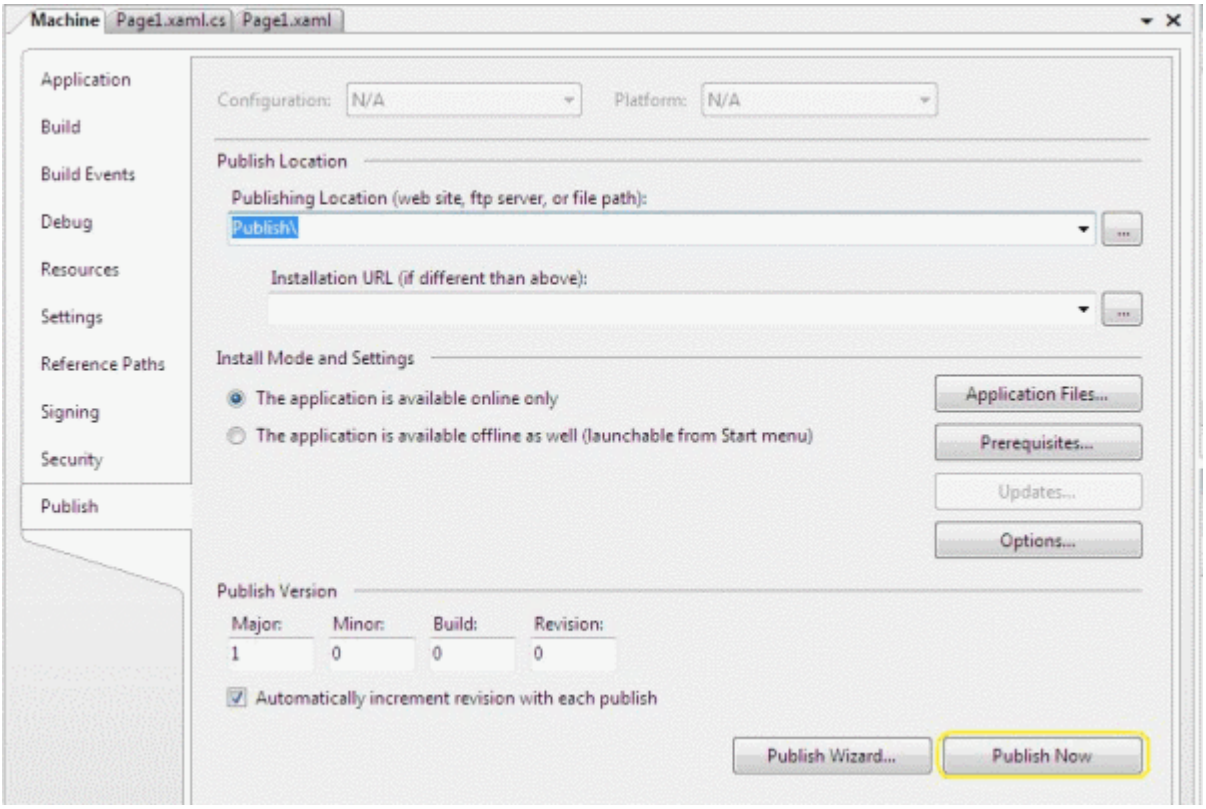
In dem Close-Ereignis des Fensters werden die Verbindungen wieder mit der Methode DeleteDeviceNotification() freigegeben.

```
//-----------------------------------------------------// 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) Handle
s 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
```

Das SPS Maschinenprogramm Machine_Final.pro muss auf dem Laufzeitsystem 1 laufen und das Programm kann im Internet Explorer 7 getestet werden.

9. Einbinden in den Vista Media Center

Habe Sie ihr Projekt hinreichend getestet und keinen Fehler festgestellt, dann können Sie es nun in den Media Center einbinden.
Rufen Sie wieder in Visual Studio die Projekteigenschaften auf, jedoch gehen Sie dann auf 'Publish'. Dort klicken Sie dann auf 'Publish Now'. Es wird nun die xbap-Datei erstellt die Sie später im Media Center aufrufen. Diesen Schritt müssen Sie immer machen, wenn Sie ihr Programm verändert haben und die Änderung auch in den Media Center übernommen werden soll.



Gehen Sie jetzt in einen Texteditor, z.B. Notepad geben Sie folgendes ein:

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

Speichern Sie es unter: 'C:\Users\<User>\AppData\Roaming\Media Center Programs\Machine.mcl'. Wenn Sie nun Ihren Media Center starten finden Sie ihr Programm unter 'Online Media -> program library -> programs by name -> Machine'.
Dies ist die einfachste Form der Einbindung in den Windows Vista Media Center. Weitere Informationen zur Einbindung in den Media Center finden Sie hier.

10. Download Expression Blend Beispiel:

https://infosys.beckhoff.com/content/1031/tcsample_expression/Resources/12493868299.zip

5 ADS Return Codes

Gruppierung der Fehlercodes:
Globale Fehlercodes: 0x0000 [▶ 44]... (0x9811_0000 ...)
Router Fehlercodes: 0x0500 [▶ 44]... (0x9811_0500 ...)
Allgemeine ADS Fehler: 0x0700 [▶ 45]... (0x9811_0700 ...)
RTime Fehlercodes: 0x1000 [▶ 46]... (0x9811_1000 ...)

Globale Fehlercodes

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

Router Fehlercodes

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

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

Allgemeine ADS Fehlercodes

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

Hex	Dec	HRESULT	Name	Beschreibung
0x728	1832	0x98110728	ADSERR_DEVICE_LICENSESYSTEMID	Lizenzproblem: System-ID ist ungültig.
0x729	1833	0x98110729	ADSERR_DEVICE_LICENSENOTIMELIMIT	Lizenz nicht zeitlich begrenzt.
0x72A	1834	0x9811072A	ADSERR_DEVICE_LICENSEFUTUREISSUE	Lizenzproblem: Zeitpunkt in der Zukunft.
0x72B	1835	0x9811072B	ADSERR_DEVICE_LICENSETIMETOLONG	Lizenz-Zeitraum zu lang.
0x72C	1836	0x9811072C	ADSERR_DEVICE_EXCEPTION	Exception beim Systemstart.
0x72D	1837	0x9811072D	ADSERR_DEVICE_LICENSEDUPLICATED	Lizenz-Datei zweimal gelesen.
0x72E	1838	0x9811072E	ADSERR_DEVICE_SIGNATUREINVALID	Ungültige Signatur.
0x72F	1839	0x9811072F	ADSERR_DEVICE_CERTIFICATEINVALID	Zertifikat ungültig.
0x730	1840	0x98110730	ADSERR_DEVICE_LICENSEOEMNOTFOUND	Public Key vom OEM nicht bekannt.
0x731	1841	0x98110731	ADSERR_DEVICE_LICENSERESTRICTED	Lizenz nicht gültig für diese System.ID.
0x732	1842	0x98110732	ADSERR_DEVICE_LICENSEDEMODENIED	Demo-Lizenz untersagt.
0x733	1843	0x98110733	ADSERR_DEVICE_INVALIDFNCID	Funktions-ID ungültig.
0x734	1844	0x98110734	ADSERR_DEVICE_OUTOFRANGE	Außerhalb des gültigen Bereiches.
0x735	1845	0x98110735	ADSERR_DEVICE_INVALIDALIGNMENT	Ungültiges Alignment.
0x736	1846	0x98110736	ADSERR_DEVICE_LICENSEPLATFORM	Ungültiger Plattform Level.
0x737	1847	0x98110737	ADSERR_DEVICE_FORWARD_PL	Kontext – Weiterleitung zum Passiv-Level.
0x738	1848	0x98110738	ADSERR_DEVICE_FORWARD_DL	Kontext – Weiterleitung zum Dispatch-Level.
0x739	1849	0x98110739	ADSERR_DEVICE_FORWARD_RT	Kontext – Weiterleitung zur Echtzeit.
0x740	1856	0x98110740	ADSERR_CLIENT_ERROR	Clientfehler.
0x741	1857	0x98110741	ADSERR_CLIENT_INVALIDPARM	Dienst enthält einen ungültigen Parameter.
0x742	1858	0x98110742	ADSERR_CLIENT_LISTEMPTY	Polling-Liste ist leer.
0x743	1859	0x98110743	ADSERR_CLIENT_VARUSED	Var-Verbindung bereits im Einsatz.
0x744	1860	0x98110744	ADSERR_CLIENT_DUPLINVOKEID	Die aufgerufene ID ist bereits in Benutzung.
0x745	1861	0x98110745	ADSERR_CLIENT_SYNCTIMEOUT	Timeout ist aufgetreten – Die Gegenstelle antwortet nicht im vorgegebenen ADS Timeout. Die Routeneinstellung der Gegenstelle kann falsch konfiguriert sein.
0x746	1862	0x98110746	ADSERR_CLIENT_W32ERROR	Fehler im Win32 Subsystem.
0x747	1863	0x98110747	ADSERR_CLIENT_TIMEOUTINVALID	Ungültiger Client Timeout-Wert.
0x748	1864	0x98110748	ADSERR_CLIENT_PORTNOTOPEN	Port nicht geöffnet.
0x749	1865	0x98110749	ADSERR_CLIENT_NOAMSADDR	Keine AMS Adresse.
0x750	1872	0x98110750	ADSERR_CLIENT_SYNCINTERNAL	Interner Fehler in Ads-Sync.
0x751	1873	0x98110751	ADSERR_CLIENT_ADDHASH	Überlauf der Hash-Tabelle.
0x752	1874	0x98110752	ADSERR_CLIENT_REMOVEHASH	Schlüssel in der Tabelle nicht gefunden.
0x753	1875	0x98110753	ADSERR_CLIENT_NOMORESYM	Keine Symbole im Cache.
0x754	1876	0x98110754	ADSERR_CLIENT_SYNCRESINVALID	Ungültige Antwort erhalten.
0x755	1877	0x98110755	ADSERR_CLIENT_SYNCPORTLOCKED	Sync Port ist verriegelt.
0x756	1878	0x98110756	ADSERR_CLIENT_REQUESTCANCELLED	Die Anfrage wurde abgebrochen.

RTime Fehlercodes

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

Hex	Dec	HRESULT	Name	Beschreibung
0x1018	4120	0x98111018	RTERR_VMXDISABLED	Intel VT-x Erweiterung ist nicht aktiviert im BIOS.
0x1019	4121	0x98111019	RTERR_VMXCONTROLSMISSING	Fehlende Funktion in Intel VT-x Erweiterung.
0x101A	4122	0x9811101A	RTERR_VMXENABLEFAILS	Aktivieren von Intel VT-x schlägt fehl.

Spezifische positive HRESULT Return Codes:

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

TCP Winsock-Fehlercodes

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

Mehr Informationen:
www.beckhoff.de/tx1000

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