# Creation of .NET Controls

WinCC Professional / V15/Runtime System / Visual Studio

**SIEMENS**
*Ingenuity for life*

**Siemens Industry Online Support**

# Legal information

**Use of application examples**

Application examples illustrate the solution of automation tasks through an interaction of several components in the form of text, graphics and/or software modules. The application examples are a free service by Siemens AG and/or a subsidiary of Siemens AG ("Siemens"). They are non-binding and make no claim to completeness or functionality regarding configuration and equipment. The application examples merely offer help with typical tasks; they do not constitute customer-specific solutions. You yourself are responsible for the proper and safe operation of the products in accordance with applicable regulations and must also check the function of the respective application example and customize it for your system.

Siemens grants you the non-exclusive, non-sublicensable and non-transferable right to have the application examples used by technically trained personnel. Any change to the application examples is your responsibility. Sharing the application examples with third parties or copying the application examples or excerpts thereof is permitted only in combination with your own products. The application examples are not required to undergo the customary tests and quality inspections of a chargeable product; they may have functional and performance defects as well as errors. It is your responsibility to use them in such a manner that any malfunctions that may occur do not result in property damage or injury to persons.

**Disclaimer of liability**

Siemens shall not assume any liability, for any legal reason whatsoever, including, without limitation, liability for the usability, availability, completeness and freedom from defects of the application examples as well as for related information, configuration and performance data and any damage caused thereby. This shall not apply in cases of mandatory liability, for example under the German Product Liability Act, or in cases of intent, gross negligence, or culpable loss of life, bodily injury or damage to health, non-compliance with a guarantee, fraudulent non-disclosure of a defect, or culpable breach of material contractual obligations. Claims for damages arising from a breach of material contractual obligations shall however be limited to the foreseeable damage typical of the type of agreement, unless liability arises from intent or gross negligence or is based on loss of life, bodily injury or damage to health. The foregoing provisions do not imply any change in the burden of proof to your detriment. You shall indemnify Siemens against existing or future claims of third parties in this connection except where Siemens is mandatorily liable.

By using the application examples you acknowledge that Siemens cannot be held liable for any damage beyond the liability provisions described.

**Other information**

Siemens reserves the right to make changes to the application examples at any time without notice. In case of discrepancies between the suggestions in the application examples and other Siemens publications such as catalogs, the content of the other documentation shall have precedence.

The Siemens terms of use (https://support.industry.siemens.com) shall also apply.

**Security information**

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected to an enterprise network or the Internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place.

For additional information on industrial security measures that may be implemented, please visit https://www.siemens.com/industrialsecurity.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends that product updates are applied as soon as they are available and that the latest product versions are used. Use of product versions that are no longer supported, and failure to apply the latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed at: https://www.siemens.com/industrialsecurity.

# Table of contents

# 1 Introduction

## 1.1 Overview

This purpose of this application example is to show you an example of how Microsoft Visual Studio creates a ".NET Control".
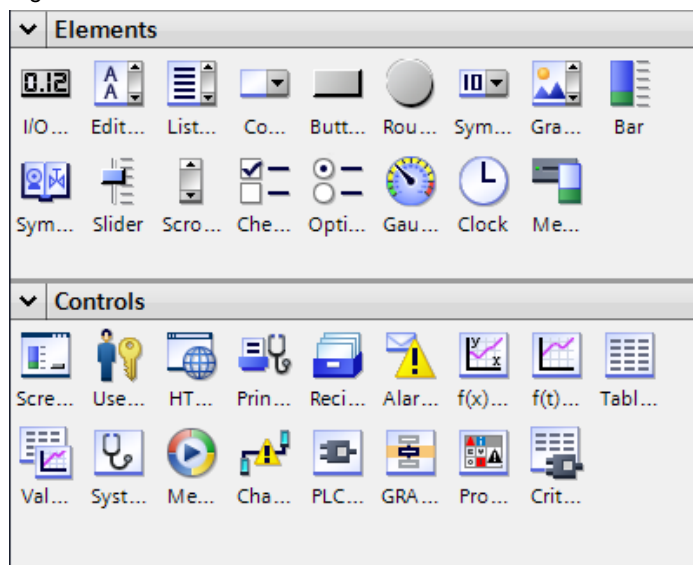
We will also show you how to install the .NET Control

- in the WinCC Professional Engineering Station.
- in the WinCC  Professional Runtime Station.

**WinCC Professional Controls and Elements**

The standard "Controls and Elements" for WinCC Professional are depicted in the following figure.

Fig. 1-1



WinCC Professional allows you to create customer-specific controls and install and use them on other WinCC Runtime Professional stations.
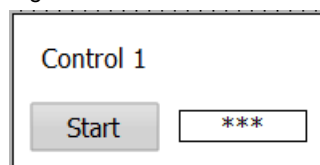
**.Net Control to be created**

The following figure depicts the .NET Control to be created, where the individual configuration steps are shown.

The Control consists of

- A title
- A button
- An I/O field

Fig. 1-2

## 1.2 Functional description

The following functions can be executed using the finished .NET Control.

- A system function that has been freely selected from "WinCC Professional" can be executed using the "Start" button.
- The I/O field can be assigned any variable from "WinCC Professional".

**Required knowledge**

The application example requires the following basic knowledge:

- Making projects with WinCC Professional
  Basics are taught in the WinCC Professional SITRAIN course.
  See Entry ID [109758618](#).
- Microsoft Visual Studio
- C# Programming
- Microsoft .NET Framework and .NET Managed Code Assemblies

## 1.3 Components used

The application example was created with the following hardware and software components:

Table 1-1

| Component | Quantity | Article number | Note |
|---|---|---|---|
| WinCC Professional V15 | 1 | 6AV2103-0AA05-0AA7 | Or later version |
| WinCC Runtime Professional V15 | 1 | 6AV2103-0DA05-0AA5 | Or later version |
| Microsoft Visual Studio 2015 | 1 | Refer to the Internet | Or later version |
| Microsoft Windows SDK Tools | 1 | Refer to the Internet | Free Microsoft download. |

This application example consists of the following components:

Table 1-2

| Component | File name | Note |
|---|---|---|
| Example project/files | 109759944_Prepare.NETControls_CODE | V1.0 |
| Documentation | 109759944_Prepare.NETControls_DOC.docx | V1.0 |

# 2 Engineering

## 2.1 Configuration and implementation of .NET controls

In this chapter, there is an example of how to create .NET Controls in Visual Studio.

| Note | If you don't have much experience of the subject and are not familiar with the software tool "gacutil.exe" for example, then you should first read chapter 3 "Useful information". |
|------|------|

**.NET Version**

Before you start to configure the .NET Control, check the .NET version that is installed already. The version used must be the same on all computers and must match the version with which the .NET Control was created. Open the control panel from the PC and from there open the category "Programs and Functions" to determine which .NET version is installed.
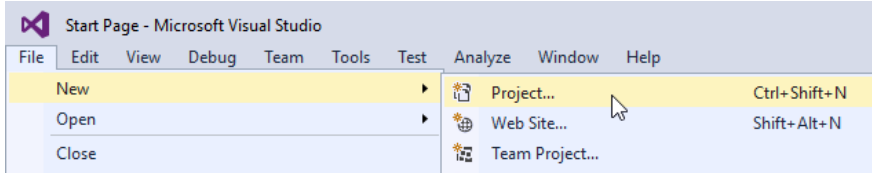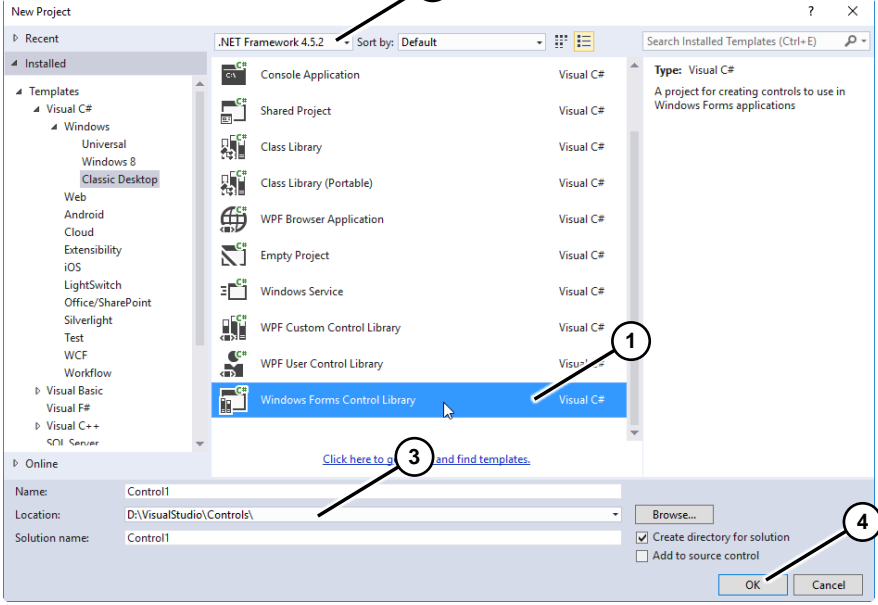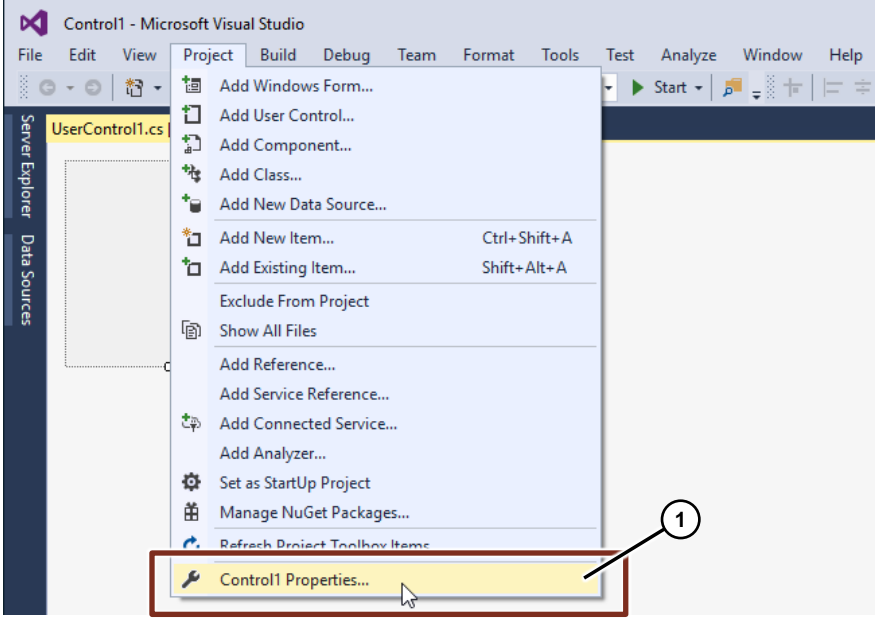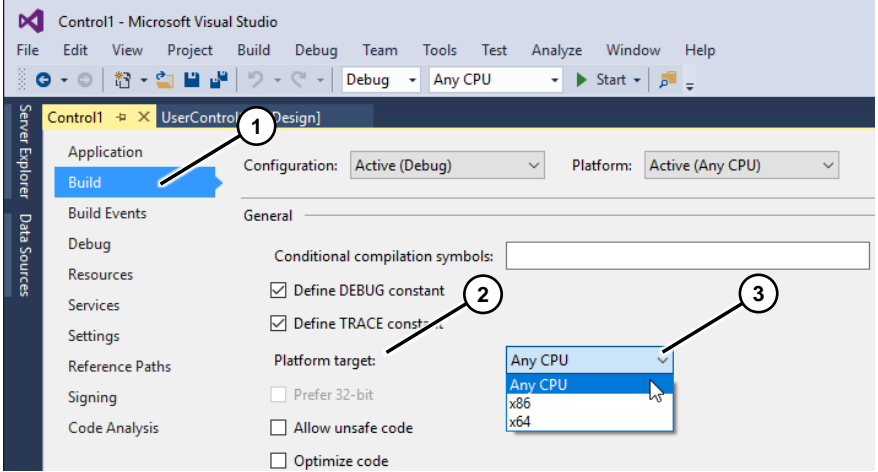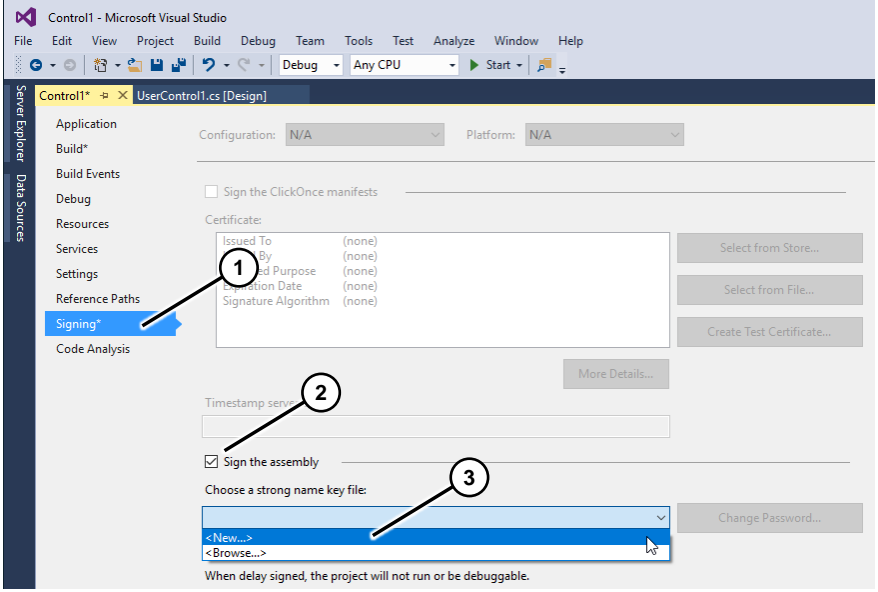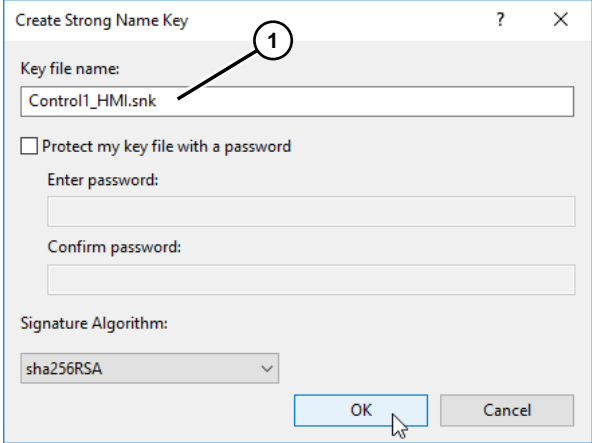
Fig. 2-1
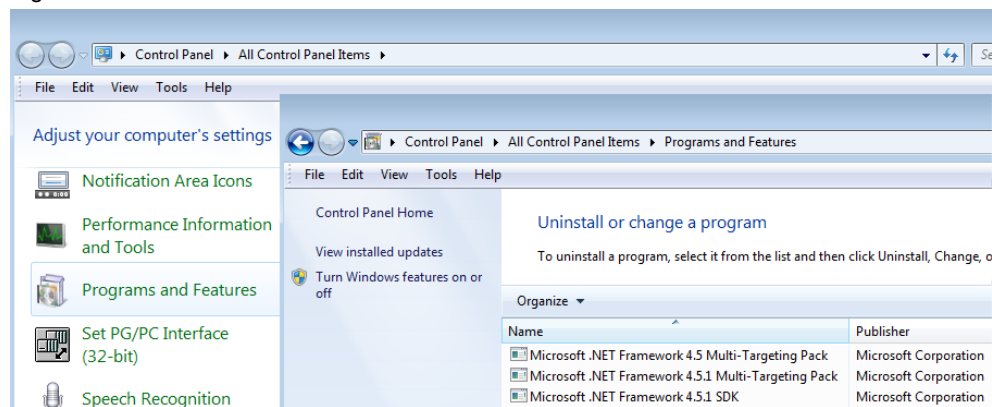


**Create .NET Control**

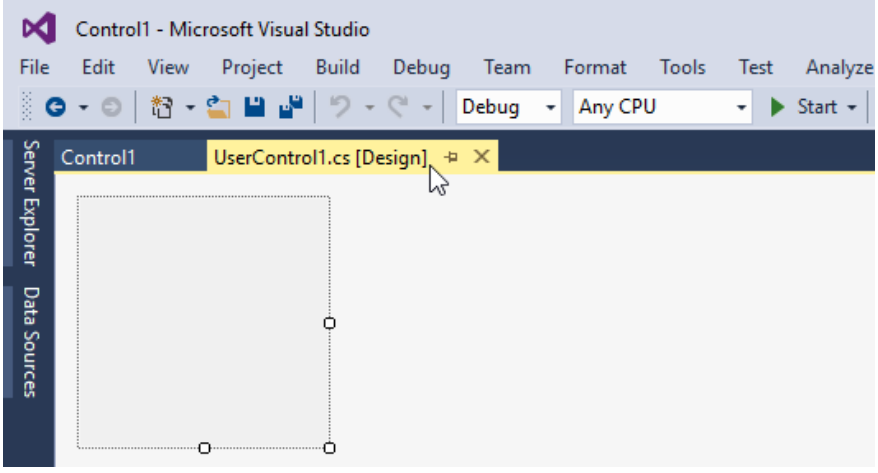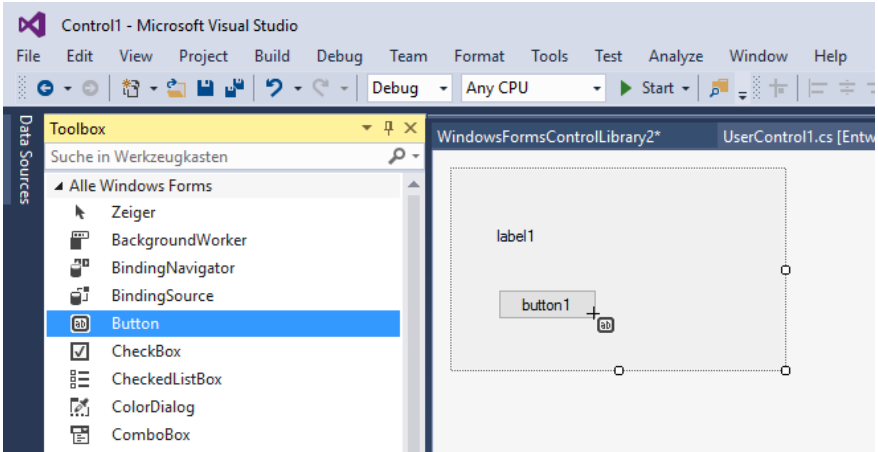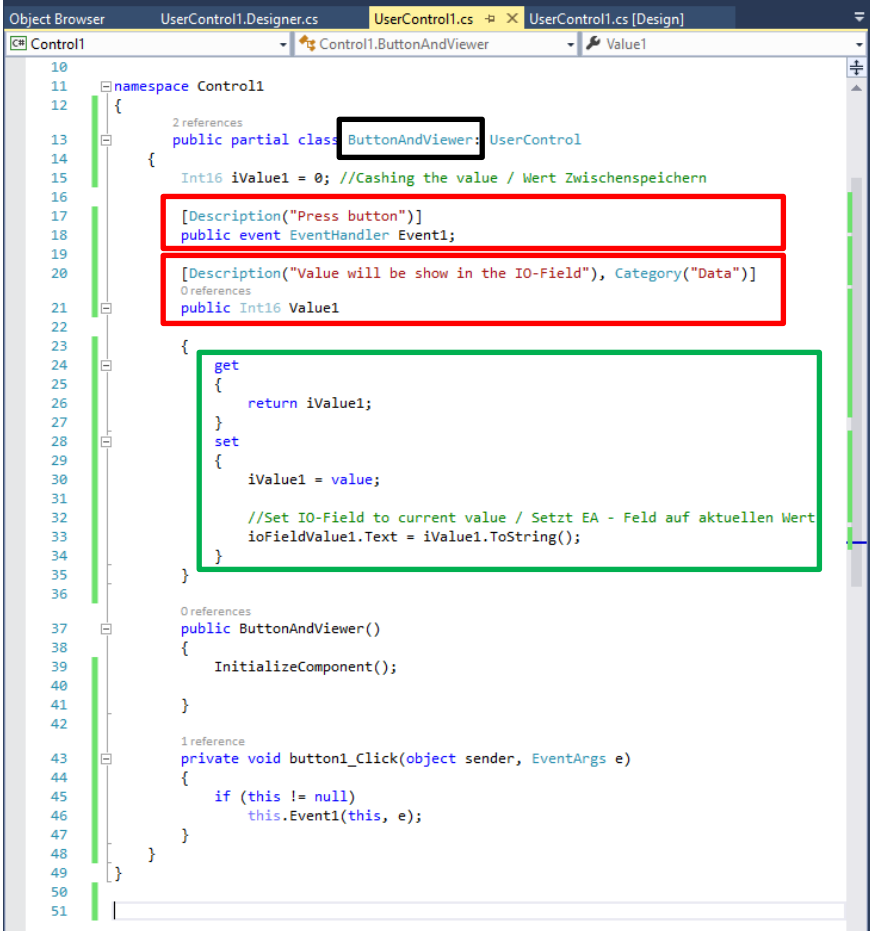The following table shows you how to create the .NET Control.

Table 2-1

| No. | Description |
|-----|-------------|
| 1. | **Create new Visual Studio project**<br><br>• Open Visual Studio<br>• Select "File> New > Project…"<br>This opens the "New Project" dialog window.<br><br> |

| No. | Description |
|-----|-------------|
| 2. | **"New Project" dialog window**<br><br>• Select a template (1).<br>  "Installed > Templates > Visual C# > Windows > Classic Desktop ><br>  Windows Forms Control Element Library"<br>• Use the drop-down list to specify the .NET Framework version (2).<br>  .NET Framework 4.5.2 is used in this example.<br>• Enter a name and the file path (3).<br>  In this example: "D:\Visual Studio\Controls\Control1".<br>• Confirm your entries with "Ok" (4).<br>  The new project will be created once you confirm.<br><br> |

| No. | Description |
|---|---|
| 3. | **Project properties**<br><br>• Use the menu bar to open the properties of the newly created project "Control1"<br>   - "Project > Control1 Properties…" (1).<br>     This opens the "Control1 Properties" menu window.<br><br> |
| 4. | **"Control1 Properties" menu window**<br><br>• Select "Create" (1) in the menu window on the left-hand side.<br>• Open the drop-down list next to "Target Platform" (2) and select the option "Any CPU" (3).<br><br> |

| No. | Description |
|---|---|
| 5. | **"Control1 Properties" menu window**<br><br>• Select the "Signing" option in the menu window on the left-hand side (1).<br>• Check the "Sign assembly" box (2).<br>• Open the drop-down list under "Select key file with strong name:" and select the "New ..." option (3).<br>A dialog window opens.<br><br>**Note**<br>Assemblies must be labeled with an informative key name (a unique name) before they can be installed in the GAC.<br><br> |
| 6. | **"Create key for a strong name" dialog**<br><br>• Enter a name in the "Key file name" field. In this case, it's "Control1_HMI.snk".<br>The name extension is "**\*.snk**" (1).<br>• Confirm the entry with "OK".<br><br> |

| No. | Description |
|---|---|
| 7. | **Call up the "UserControl1" tab.**<br><br>Change from the "Control1" tab to the "UserControl1.cs" tab.<br><br> |
| 8. | **Adding elements**<br><br>A "title, an I/O field, and a button" are used for this application example (see Fig. 1-2).<br><br>• Open the "Toolbox" (View > Toolbox) and add the following objects.<br>To do this, select the corresponding object in the "Toolbox" and drag it into the "UserControl1" using "Drag&Drop".<br><br>    ButtonAndViewer            (User Control)<br>- "Label"         Name:titleLabel1    (Title)<br>- "Button"       Name: buttonEvent1  (Button)<br>- "Textbox"     Name: ioFieldValue1  (I/O field)<br><br> |

| No. | Description |
|---|---|
| 9. | **Configuration of controls (properties and events)** |
|  | • Open the code of the control (View > Code) This opens a predetermined program code that must be extended according to the required task. |



**Notes**

Area highlighted in black

• The name of the control should be unique (in this case, it's "ButtonAndViewer"). The name "UserControl1" is not preferable, for example.
The name can be changed under the "UserControl.cs" tab.

Area highlighted in red

• You have to add a "Description" to the variable to detect the "Event1" variable parameter in the WinCC TIA Portal project.

• You have to add a "Category" to the variable to detect the "Value1" variable parameter in the WinCC TIA Portal project. In this example, it's the "Data" category.

Area highlighted in green

• To read or set a variable value externally in the TIA Portal, the variable must be declared as "public" and provided with "getter" and "setter" methods.

| No. | Description |
|---|---|
| 10. | **Overview of code parameters and the created DLL**<br><br>The figure depicts the code for the configured button in "C #" and the finished Control in WinCC Professional.<br><br><br><br>The figure depicts the code for the configured ES field in "C #" and the finished Control in WinCC Professional.<br><br> |

2 Engineering

| No. | Description |
|-----|-------------|
| 11. | **Adjust assembly value** |
|  | WinCC Runtime Professional always makes attempts at loading the "1.0.0.0" version of a user-defined .NET Control from the GAC or the directory to be implemented. |
|  | Open the project file to manually change the above-mentioned value. The value under [assembly: AssemblyVersion()] can be changed to "1.0.0.0" in the "AssemblyInfo.cs" directory. |

```
31    //
32    // You can specify all the values or you can (
33    // by using the '*' as shown below:
34    // [assembly: AssemblyVersion("1.0.*")]
35    [assembly: AssemblyVersion("1.0.0.0")]
36    [assembly: AssemblyFileVersion("1.0.0.0")]
37
```

Solution Explorer

Search Solution Explorer (Ctrl+ü)

Solution 'Control1' (1 project)
- Control1
  - Properties
    - AssemblyInfo.cs
  - References
  - Control1_HMI.snk
  - UserControl1.cs
    - UserControl1.Designer.cs

| No. | Description |
|---|---|
| 12. | **Create Control (create DLL file)**<br><br>• Click on "Debug > Start Debugging F5" in the menu bar (1). Make sure that the project is transferred without errors.<br><br>The created "Control1.dll" is saved in the project directory. In this example: "D:\Visual Studio\NewControls\Control1\Control1\ bin\Debug\ **Control1.dll**".<br><br>**Note**<br>Close all applications that are already using the .NET Control (DLL file) before generating. |

## 2.2 Install .NET Control on multiple PCs

**Check the dependencies**

If the created .NET Control is to be installed on multiple PCs, the DLL name used by the .NET Control must also exist on these PCs.

In the first step, the DLL used by the created .NET Control is read on the projecting PC.

### 2.2.1 Read DLL name on the projecting PC

This example uses the "ILDASM.exe" tool to read the DLL used. The "ILDASM.exe" tool can be found under the "Windows SDKs" Windows folder, for example.
(C:\Program Files (x86) > Microsoft SDKs > Windows > v"SDK Version" > bin > NETFX 4.6 Tools > ildasm.exe).

Fig. 2-2
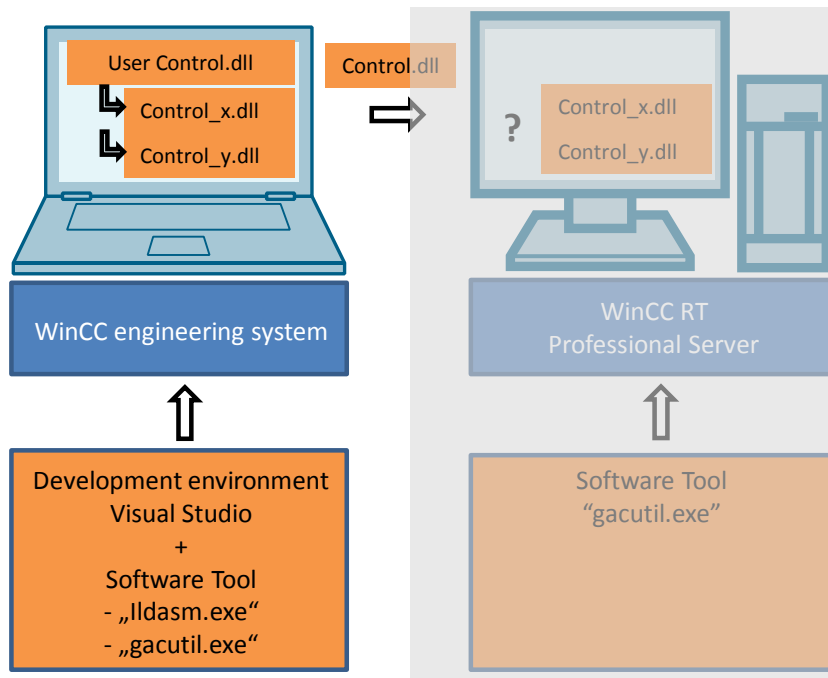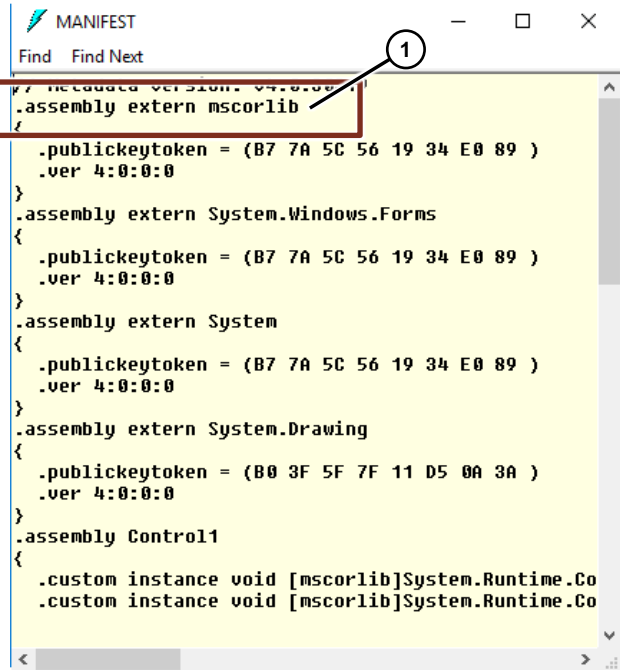


| Note | If there is no folder named "Windows SDKs" on the PC, install the "Windows (your version) SDK" option on the PC. |
| | The software can be downloaded from the Microsoft Support sites. |

### Use the "ILDASM.exe" tool

Table 2-2

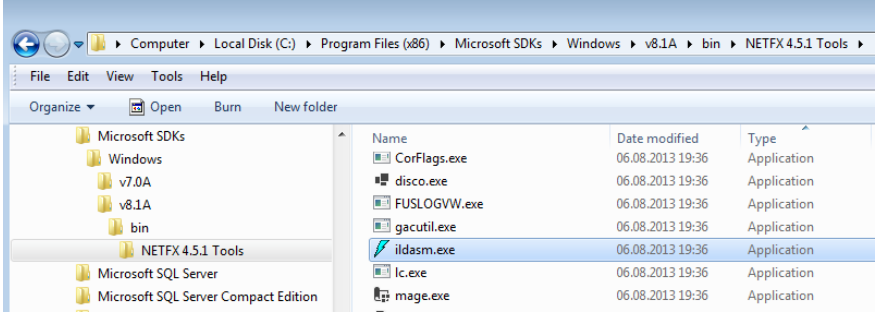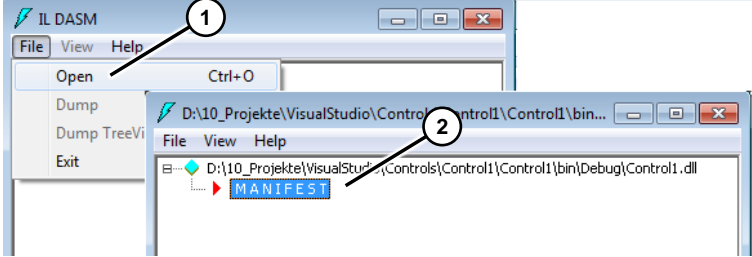| No. | Description |
|---|---|
| 1. | **Preparation**<br><br>If you want to check the requirements for the .NET Control created on a PC, then you have to save the created .NET Control in a folder on the PC to be checked. |
| 2. | **Open the "ILDASM.exe" tool**<br><br>• Double-click on the "ILDASM.exe" file to open it.<br>C:\Programme (x86) > Microsoft SDKs > Windows > v(SDK Version) > bin > NETFX (Version) Tools > ildasm.exe".<br>• This opens the "IL DASM" dialog window.<br><br>**Note**<br>Make sure that the "ildasm.exe" used is from the "Application" type.<br><br> |
| 3. | **Run ILDASM.exe**<br><br>• Click on "File > Open" in the menu bar (1)<br>• Navigate to the .NET Control. In this case, it's "Control1".<br>• Double-click on "M A N I F E S T" to start the evaluation (2).<br>This opens the window with the evaluation.<br><br> |

| No. | Description |
|-----|-------------|
| 4. | **Evaluation** <br><br> Check the entries under "**external** ….." (1) <br> In this example, the DLL <br> • mscorlib <br> • System.Windows.Forms <br> • System <br> • System.Drawing <br> from the created .NET Control "Control1" is used. <br> The specified DLL must exist on all PCs where the created .NET Control "Control1" should be installed. <br><br> **Note** <br> Depending on the scope of the created "DLL", there may be additional entries in the list. <br><br>  |

### 2.2.2 Check existing DLL names on PCs

In the previous chapter, the DLL names used by the .NET Control were read using the "ILDASM.exe" tool.
In this chapter, how to specifically search for DLL names on PCs is explained. The "gacutil.exe" tool is used for this purpose.
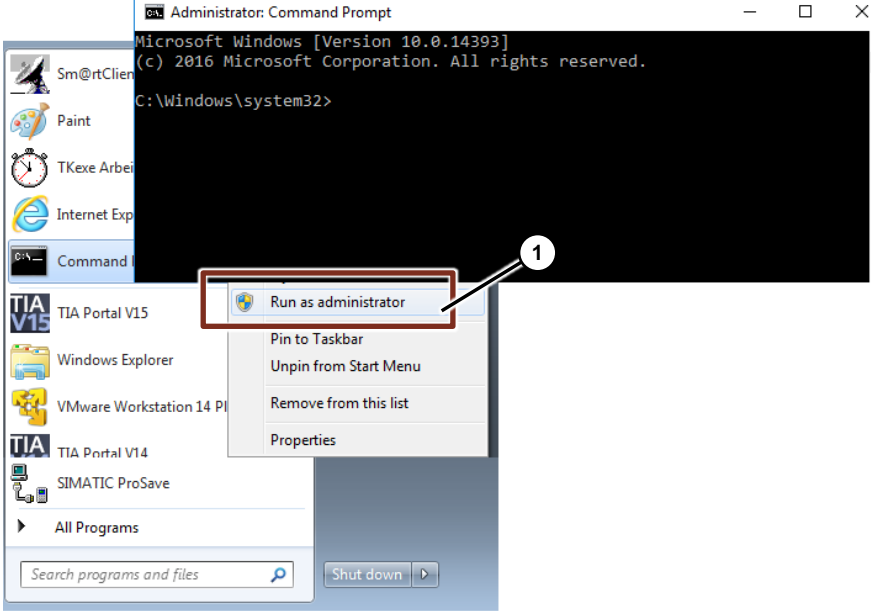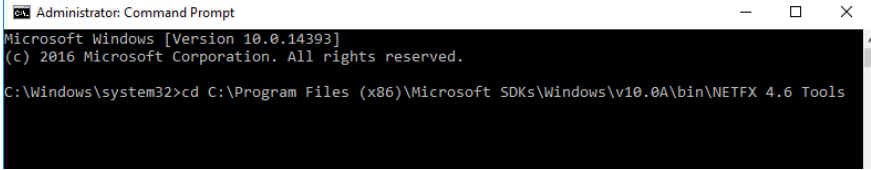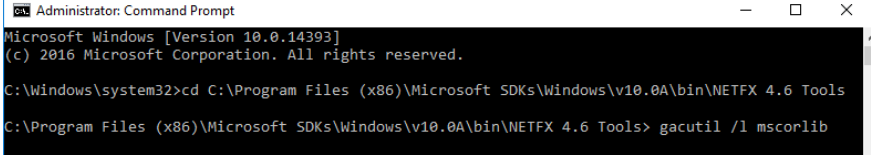
Fig. 2-3
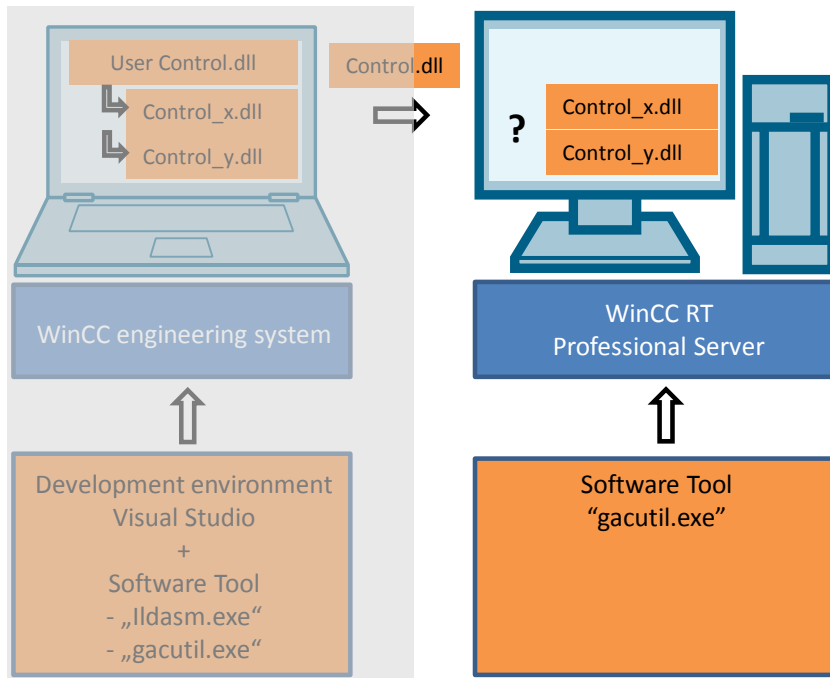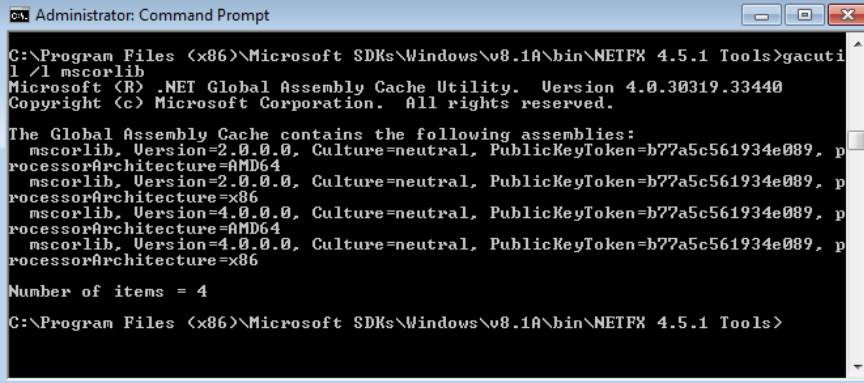


**Use the "gacutil.exe" tool**

Table 2-3

| No. | Description |
|---|---|
| 1. | **Call up the "gacutil.exe" file path**<br><br>• Open the file path for the "gacutil.exe" file.<br>"C:\Programme (x86)\Microsoft SDKs\Windows\v(SDK Version)\bin\NETFX (Version) Tools\gacutil.exe".<br>• Copy the file path from the command line (1).<br><br>**Note**<br>Make sure that the "gacutil.exe" used is from the "Application" type.<br><br> |

| No. | Description |
|---|---|
| 2. | **Call up command prompt**<br><br>• Call up the "Command Prompt" using the Windows start menu. Right-click the "Command Prompt" application and run the function as "Run Administrator" to do this (1).<br><br> |
| 3. | **Run the "gacutil.exe" tool**<br><br>• Paste the file path that you copied previously from the "gacutil.exe" tool into the command prompt window. Note the prefixed DOS command "**cd**" (change directory).<br><br>=> "**cd** C:\Programme (x86)\ Microsoft SDKs\Windows\v(SDK Version)\ bin\NETFX (Version) Tools".<br><br><br><br>• Enter the "**gacutil /L**" command and the DLL name to be checked into the working directory.<br><br>=> gacutil /L mscorlib<br><br><br><br>• Complete your entry with the "Enter" button. A list of versions of the file you are looking for is listed. |

| No. | Description |
|---|---|
| | If the DLL name is **not** found, "Number of items = 0" will be displayed instead. |



## 2.3 Installing the .NET Control on the target computers

There are several options for installing the .NET Control on the projecting PC or on an external PC.

- Installing the .NET Control in the Assembly Cash with the "gacutil.exe" tool. This variant installation will be described in further detail below. We recommend you to use this variant.

Table 2-4

| Advantage | Disadvantage |
|---|---|
| Faster access using the WinCC (TIA Portal). | Complicated installation process. |
| Uses the .Net Framework Standard Assembly Management. | 32-bit (x86) Control are not available for WinCC RT Professional. |
| The control is available for all applications. | |

- Manual installation. This variant installation will be described in further detail below.

Table 2-5

| Advantage | Disadvantage |
|---|---|
| Simple installation process. | Control is only available for applications in this installation directory (WinCC RT Professional). |
| Works for 32-bit (x86) and "any CPU" targeted controls | Does not use the .Net Framework Standard Assembly Management. |

- Installation with the "Installer Tool" (Microsoft application – not part of Visual Studio and not described in this application example).

### 2.3.1 Write .NET Control in the global "Assembly Cache"

The "gacutil.exe" tool writes the .NET Control in the global
"Assembly cache" (GAC). The global "Assembly cache" (GAC) makes .NET
Control available to all applications.
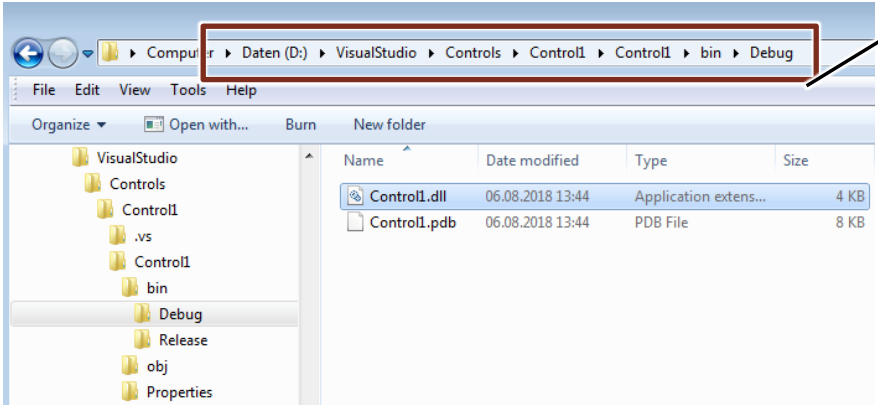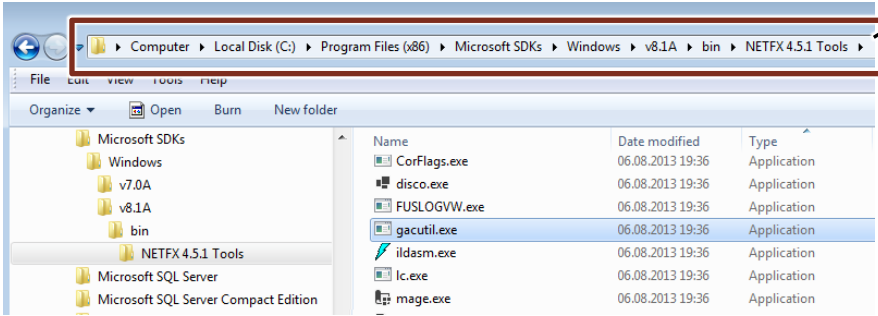
Fig. 2-4
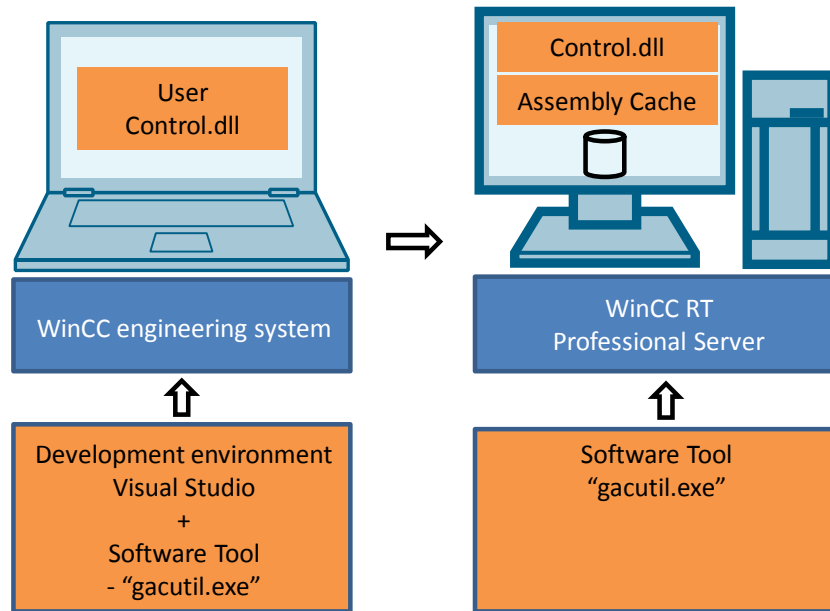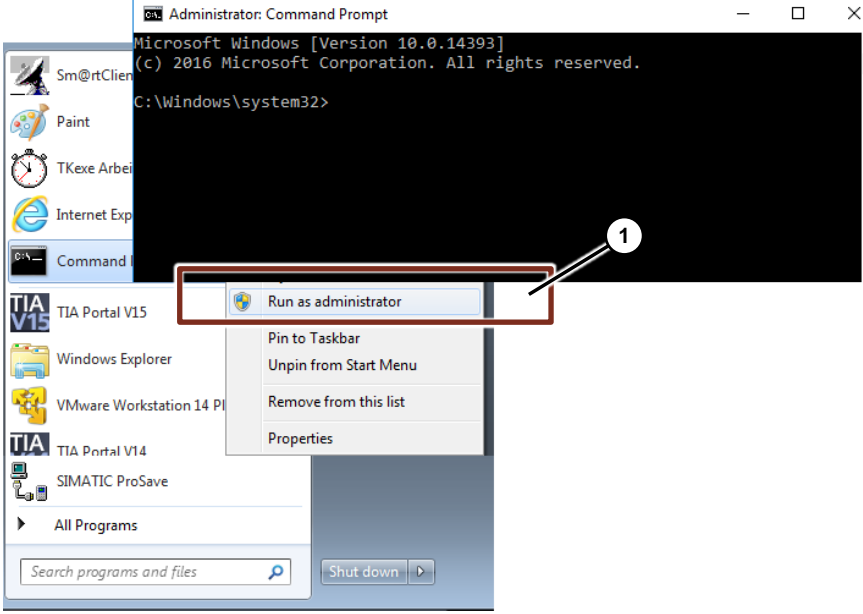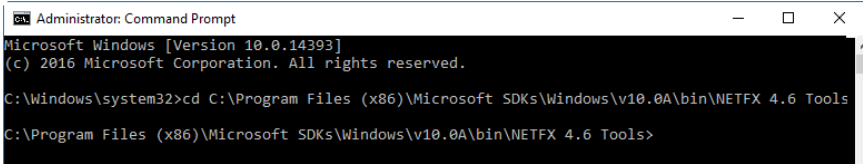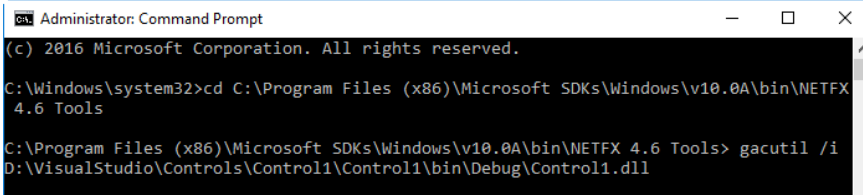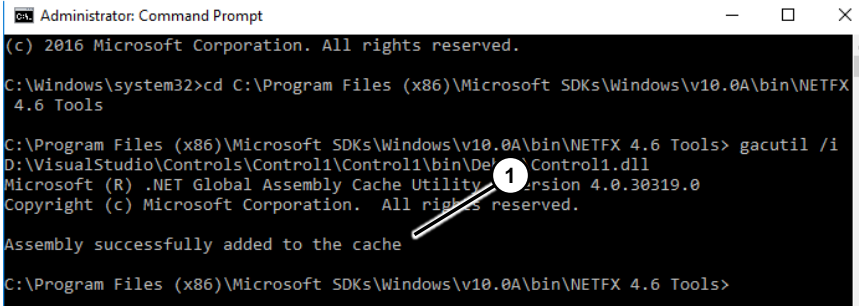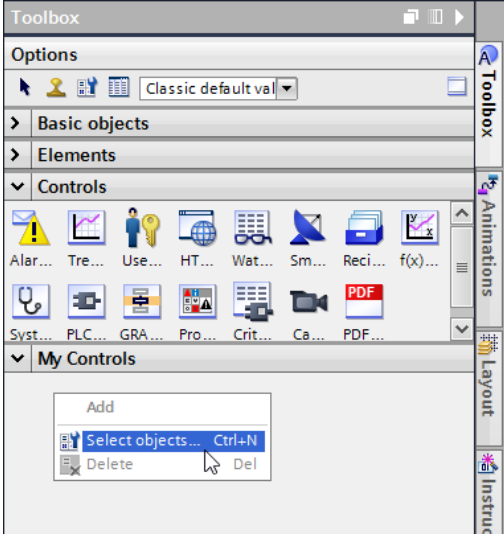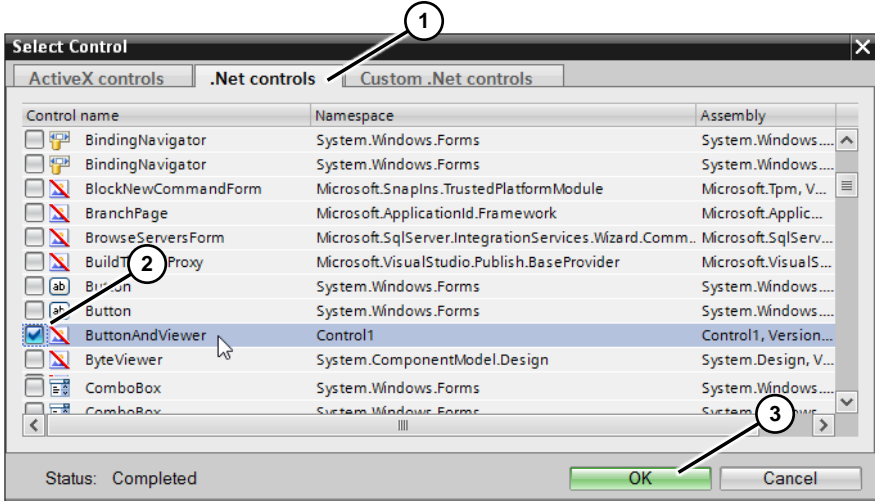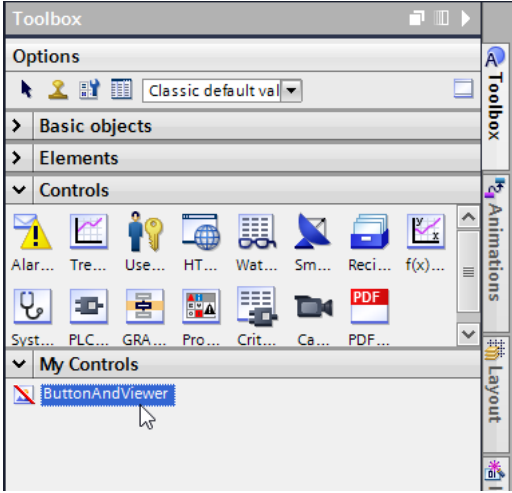
| Notes | • 32-bit (x86) Controls are not available for WinCC RT Professional. |
|---|---|
| | • The procedure assumes that you have an up-to-date version of Windows SDK Tools installed on your target computer. |
| | • GAC = An area managed by Windows for storing all .NET Framework assemblies (DLL). |
| | • All user-defined Controls with the version number "1.0.0.0" are found by WinCC RT Professional in this area. |

Table 2-6

| No. | Description |
|-----|-------------|
| 1. | **Call up the file path of the created .NET Control**<br><br>• Open the file path where the created .NET Control is saved.<br>  In this example:<br><br>  "D:\VisualStudio\Controls\Control1\Control1\bin\Debug\" (1).<br><br><br><br>**Note**<br>You can choose the storage location yourself. |
| 2. | **Call up the "gacutil.exe" file path**<br><br>• Open the file path for the "gacutil.exe" file.<br><br>  "C:\Programme (x86)\Microsoft SDKs\Windows\v(SDK Version)\bin\<br>  NETFX (Version) Tools\gacutil.exe".<br>• Copy the file path from the command line (1).<br><br>**Note**<br>Make sure that the "gacutil.exe" used is from the "Application" type.<br><br> |

| No. | Description |
|---|---|
| 3. | **Call up command prompt**<br><br>• Call up the "Command Prompt" using the Windows start menu. Right-click the "Command Prompt" application and run the function as "Run Administrator" to do this (1).<br><br> |
| 4. | **Run the "gacutil.exe" tool**<br><br>• Paste the file path that you copied previously from the "gacutil.exe" tool into the command prompt window. Note the prefixed DOS command "**cd**" (change directory).<br><br>=> "**cd** C:\Programme (x86)\ Microsoft SDKs\Windows\v(SDK Version)\ bin\NETFX (Version) Tools".<br><br><br><br>• Enter the "**gacutil /i**" command as well as the full file path and file name of the created .NET Control in the working directory (see table section no. 1).<br><br>=> gacutil /i D:\VisualStudio\Controls\Control1\Control1\bin\ Debug\Control1.dll<br><br> |

| No. | Description |
|---|---|
| | • Confirm your entry with the "Enter" button.<br>The following message appears on the PC.<br>(Assembly successfully added to the cache (1)).<br><br><br><br>The installation is then complete. |
| 5. | **Note**<br>If there are spaces in the .NET Control path specification, set the path specification to "superscript".<br>=> gacutil /i **"**D:\Visual Studio\Controls HMI\Control1\Control1\bin\ Debug\Control1.dll**"** |
| 6. | **Call up .NET Control in WinCC Professional**<br><br>• Open the WinCC Professional project.<br>• Open the "Tools" task card.<br>• Open the "My Controls" spectrum. This opens the "Select control" dialog window.<br><br> |

| No. | Description |
|---|---|
| 7. | **"Select control" dialog window**<br><br>- Select the ".NET Controls" register (1)<br>- Select the check box next to .NET Control (2).<br>- Confirm your entries with "OK" (3).<br><br> |
| 8. | **.NET Control view**<br><br>The created .NET Control is displayed in the "My Controls" spectrum and can be inserted into the system display using "Drag&Drop".<br><br> |

## 2.3.2 .NET Control manual installation

Figure 2-5
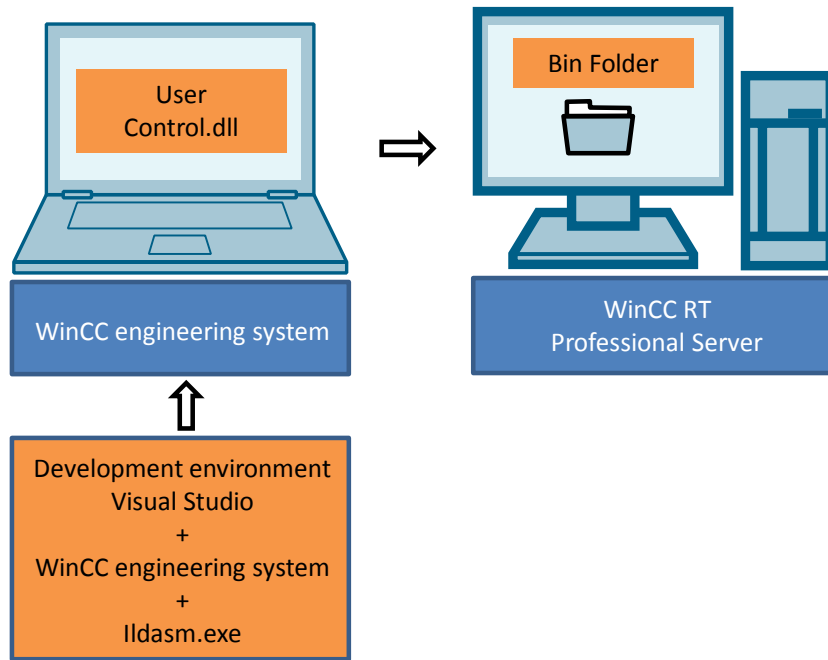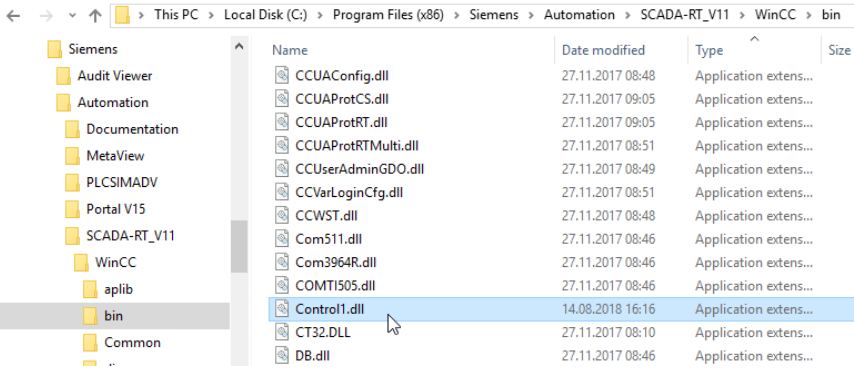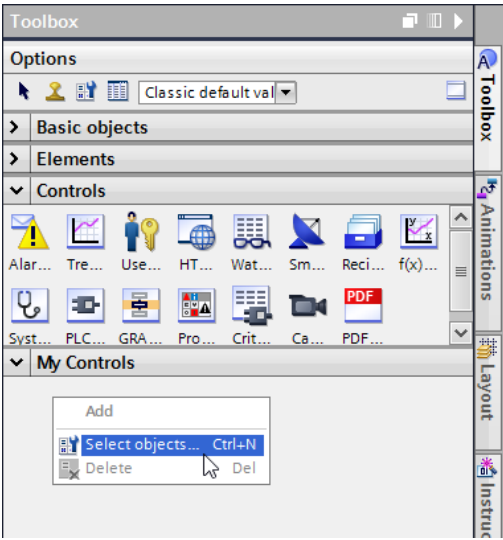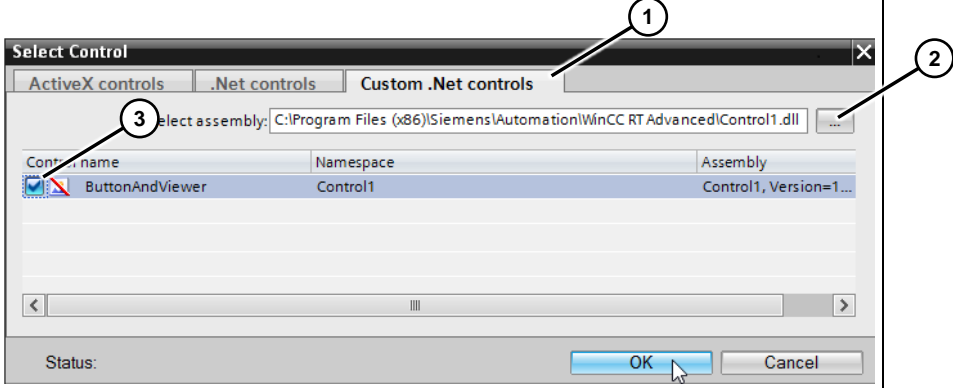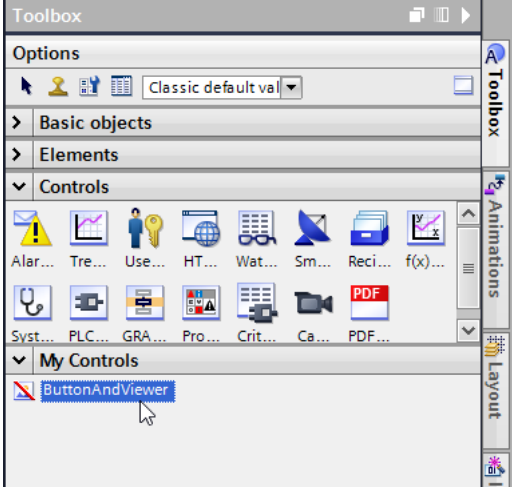
**Incorporate .NET Control in WinCC Professional Runtime**

Manually copy the "User Control.dll" into the installation path of WinCC RT Professional

Table 2-7

| No. | Description |
|-----|-------------|
| 1. | **Call up the file path of the created .NET Control**<br><br>• Open the file path where the created .NET Control is saved.<br>  In this example:<br><br>  "D:\VisualStudio\Controls\Control1\Control1\bin\Debug\" (1).<br>• Select the .NET Control and copy the file.<br><br> |

| No. | Description |
|---|---|
| 2. | **Call up the WinCC Runtime storage path** <br><br> • Open the following WinCC Professional installation path. "C:\Program (86)\Siemens\Automation\SCADA-RT_V11\WinCC\\**bin**<br> • Paste the .NET Control file that was previously copied into this directory.<br><br><br><br>**Note**<br>Always use the specified installation path. This ensures that WinCC Runtime can automatically access the "DLL". |
| 3. | **Call up .NET Control in WinCC Professional** <br><br> • Open the WinCC Professional project.<br> • Open the "Tools" task card.<br> • Open the "My Controls" spectrum. This opens the "Select control" dialog window.<br><br> |

| No. | Description |
|---|---|
| 4. | **"Select control" dialog window**<br><br>• Select the "Specific .NET Controls" register (1)<br>• Navigate to the file path where the .NET Control is saved (2).<br>• Select the .NET Control and press "open".<br>• Select the check box next to .NET Control (3).<br>• Confirm your entries with "OK".<br><br> |
| 5. | **.NET Control view**<br><br>The created .NET Control is displayed in the "My Controls" spectrum and can be inserted into the system display using "Drag&Drop".<br><br> |

### 2.3.3 "Installer Generation Add-in" for Visual Studio

You can create an installation file for your .NET Control (MSI file) using the "Installer Generation Add-in" for Visual Studio. The file may contain further installation files for installing necessary controls, objects, and libraries on the target computer, for example.
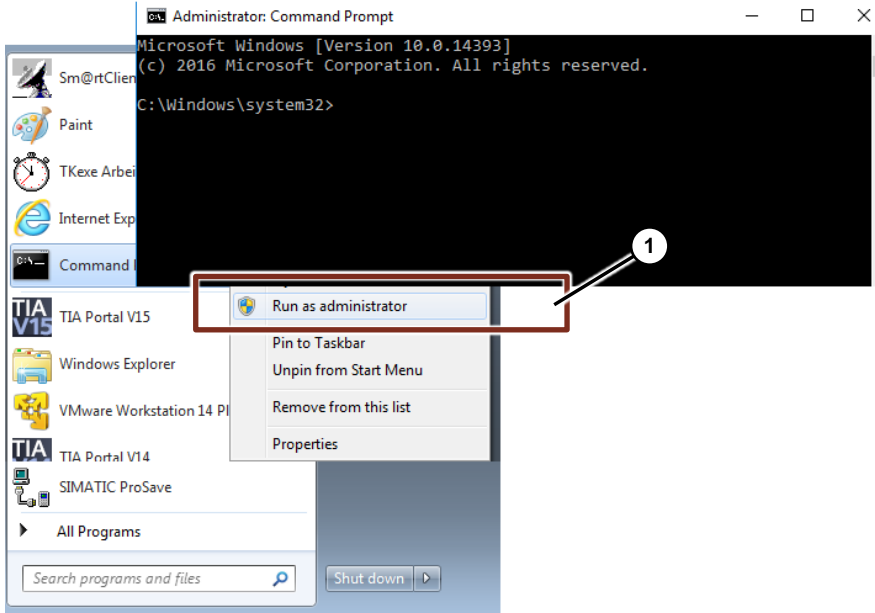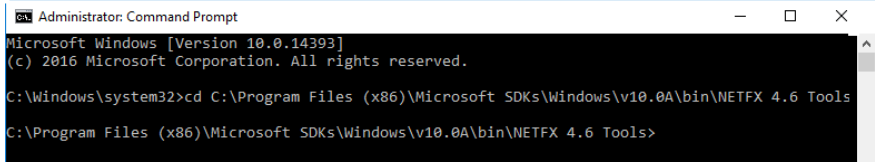
The procedure is **not** described in more detail in this application example. You can have a look at the online examples from Microsoft for more information on this.
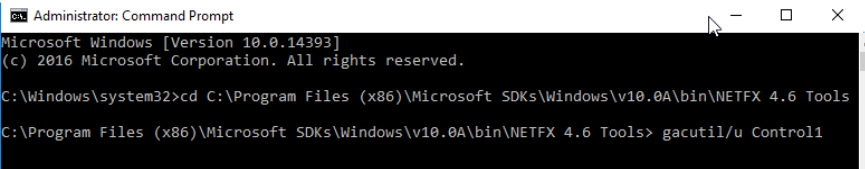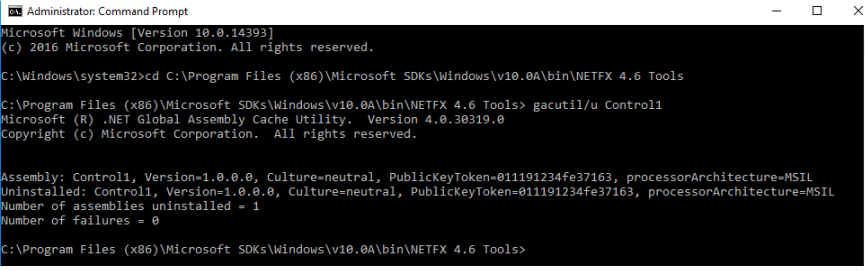
## 2.4 Uninstalling the .NET Control from a PC

| Note | Before removing the Control, close all applications that are using this Control. |
|------|------|

### 2.4.1 Uninstalling the .NET Control from Assembly Cache

Table 2-8

| No. | Description |
|-----|-------------|
| 1. | **Call up command prompt**<br><br>• Call up the "Command Prompt" using the Windows start menu. Right-click the "Command Prompt" application and run the function as "Run Administrator" to do this (1).<br><br> |
| 2. | **Run the "gacutil.exe" tool**<br><br>• Paste the file path from the "gacutil.exe" tool into the command prompt window. Note the prefixed DOS command "**cd**"<br><br>In this example:<br>=> "**cd** C:\Programme (x86)\ Microsoft SDKs\Windows\v(SDK Version)\ bin\NETFX (Version) Tools".<br><br> |

| No. | Description |
|---|---|
| | • Enter the "**gacutil /u**" command as well as the file name of the created .NET Control in the working directory.<br><br>In this example:<br>=> gacutil/u  Control1<br><br>![Command Prompt showing gacutil/u Control1 command]<br><br>• Confirm your entry with the "Enter" button.<br>The following message appears on the PC.<br><br>![Command Prompt showing gacutil uninstall result]<br><br>The uninstallation is then complete. |

### 2.4.2 Uninstalling the .NET Control from WinCC RT Professional

Call up the file path where the .NET Control is stored. Have a look at Chapter <span>2.3.2</span> "<span>.NET Control manual installation</span>" for more details.

Based on the example, the .NET Control is in the directory "C:\Program (86)\Siemens\Automation\SCADA-RT_V11\WinCC\bin".

| Note | If you delete the .NET Control, the WinCC Professional project no longer has access to the file. |
|---|---|

**Fault description**

If you use a .NET Control in a project and uninstall this Control from the installation path, then the Control can no longer be displayed. The following message/view is displayed.

Fig. 2-6
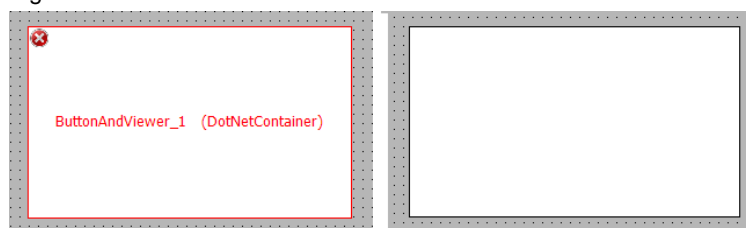
## 2.5 Error handling

- When you copy the .NET Control in the Windows "Command Prompt window", the specified .NET file is not found.
    - Check the spelling (is it with or without the ".dll" file extension?).
    - Are there spaces in the file path (put the file path into "inverted commas")?


- The .NET Control is not listed under "My Controls".
    - Check the spelling in the "Select Control" window
      (- Name Control // -Name space).
    - Check for possible error messages in the Windows "Command Prompt window".


- Visual Studio does not recognize changes when testing the .NET Control.
    - If you have already transferred the created .NET Control to the "Assembly Cache", you have to uninstall the .NET Control from the Assembly Cache first. Have a look at Chapter 2.4.1 "Uninstalling the .NET Control from Assembly Cache" for more details.

# 3 Useful information

## 3.1 Microsoft SDK

Microsoft (Version) SDK (Software Development Tool) is a free tool that you can download from the Microsoft support sites.
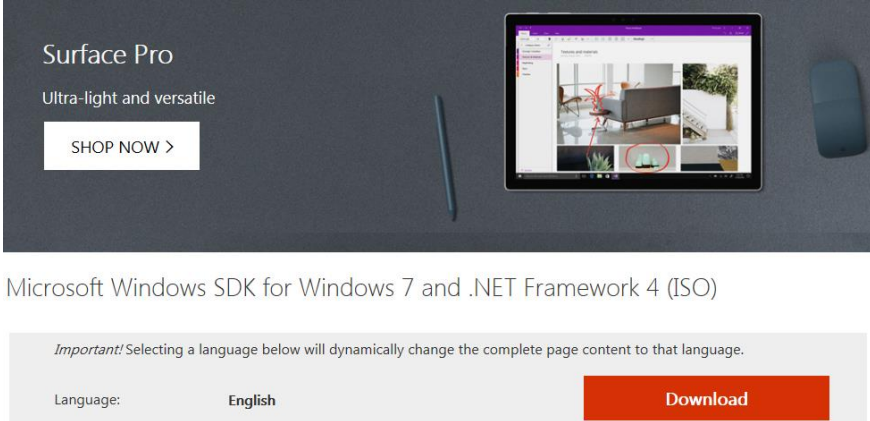
The tool contains, among other things, the two software tools

- "ILDASM.exe"
    - This example uses the "ILDASM.exe" (intermediate language disassembler) tool to read the DLL used.
- "gacutil.exe"
    - If you are searching for a DLL name on a PC, then use the "gacutil.exe" tool.
    - If you would like to write or delete the created .NET Control in the Assembly Cache, use the "gactuil.exe" tool.
    - If you manually install the .Net Control, then check that all assemblies are also present on the target computers using the .NET Control beforehand. You can use the Microsoft Windows SDK Utility gacutil.exe to check this.

| Note | "Gacutil" is automatically installed by Visual Studio, Windows SDK and other systems. |
|------|----------------------------------------------------------------------------------------|

**Installing Windows SDK**

Table 3-1

| No. | Description |
|-----|-------------|
| 1. | **Download Microsoft SDK**<br><br>• Call up the Microsoft Download Center site.<br>• Enter "Microsoft SDK" as the search term.<br>• Download the "Offline file" or alternatively you can use the "Online download".<br><br> |

| No. | Description |
|---|---|
| 2. | **Install Microsoft SDK** <br><br> • Start the installation with the "Setup.exe". <br> • Follow the set-up instructions. <br><br> ![Windows SDK Setup Wizard screenshot] <br><br> Windows® SDK for Windows® 7 and .NET Framework 4 <br><br> Windows® SDK Setup Wizard <br><br> Welcome to the Setup Wizard. <br><br> You can now install the Microsoft® Windows® Software Development Kit for Windows 7 and .NET Framework 4 <br><br> This SDK provides headers, libraries, tools, samples, VC++ 2010 compiler/CRT, and other resources to help you develop Windows applications using both native and managed technologies. This SDK supports developing on and for Windows 7, Windows Server 2008 R2, Windows Vista, Windows XP and Windows Server 2008, and targeting .NET Framework version 4 and down level versions to 2.0. |
| 3. | **Call up file path "ildasm" and "gacutil.exe"** <br><br> After installing the Microsoft SDK, the files "ildasim.exe" and "gacutil.exe" are located at the following path. <br><br> C:\Program Files (x86) > Microsoft SDKs > Windows > v"SDK Version" > bin > NETFX 4.6 Tools > |

## 3.2 Visual Studio

Visual Studio is a development environment that enables the creation, debugging, and deployment of software for Windows, Microsoft Office, and more.
The .NET Control for the WinCC (TIA Portal) is created in this example.

The creation environment is "**C#**" (C Sharp).
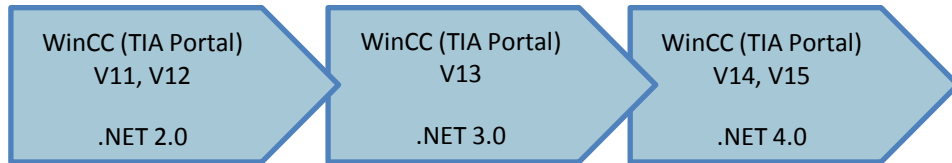
## 3.3 Assemblies / Global Assembly Cache (GAC)

• Assemblies/Assembly
An application or DLL created with the .NET Framework that requires the .NET Framework at runtime.

=> Files with the extension "*.exe" or "*.dll"

• Global Assembly Cache
The Global Assembly Cache (GAC) is a system-wide directory for assemblies.
The directory is located in the file system under " C:\Windows\assembly ".

## 3.4 .Net Framework Version

The ".NET Framework Control" used must be compatible with the TIA Engineering Version used.

Example: A WinCC (TIA Portal) V13 project cannot use a Control that was created with .NET 4.0.

Fig. 3-1

| WinCC (TIA Portal) V11, V12 | WinCC (TIA Portal) V13 | WinCC (TIA Portal) V14, V15 |
|---|---|---|
| .NET 2.0 | .NET 3.0 | .NET 4.0 |

## 3.5 Write .NET Control in the global "Assembly Cache"

The "gacutil.exe" tool writes the created .NET Control in the global Assembly Cache (GAC). The global "Assembly Cache" (GAC) makes .NET Control available to all applications.

If your Control uses the WinCC ODK API, it must be created as a 32-bit Control.

However, 32-bit Controls cannot be found under the ".Net Controls" tab if added to WinCC ES Professional. In this case, they must be added with the "Custom .Net Controls" tab. Furthermore, they must be installed in the application runtime directory, which is normally found under C:\Program Files (x86)\Siemens\Automation\SCADA-RT_V11\WinCC\bin, instead of the Global Assembly Cache.

If you also wish to use your Control on WebNavigator Clients, we recommend that you develop the Control for "Any CPU" target platform and install it in the GAC. This is the standard way to get around .Net Assemblies in Windows so that they are also compatible with the Controls in a non-WinCC environment.

Controls developed for "Any CPU" targets and installed in the GAC will appear in the ".Net Controls" tab if they have been installed on WinCC ES Professional.

# 4 Appendix

## 4.1 Service and support

**Industry Online Support**

Do you have any questions or need assistance?

Siemens Industry Online Support offers round the clock access to our entire service and support know-how and portfolio.

The Industry Online Support is the central address for information about our products, solutions and services.

Product information, manuals, downloads, FAQs, application examples and videos – all information is accessible with just a few mouse clicks:
https://support.industry.siemens.com/

**Technical Support**

The Technical Support of Siemens Industry provides you fast and competent support regarding all technical queries with numerous tailor-made offers – ranging from basic support to individual support contracts. Please send queries to Technical Support via Web form:
https://www.siemens.com/industry/supportrequest

**SITRAIN – Training for Industry**

We support you with our globally available training courses for industry with practical experience, innovative learning methods and a concept that's tailored to the customer's specific needs.

For more information on our offered trainings and courses, as well as their locations and dates, refer to our web page:
https://www.siemens.com/sitrain

**Service offer**

Our range of services includes the following:

- Plant data services
- Spare parts services
- Repair services
- On-site and maintenance services
- Retrofitting and modernization services
- Service programs and contracts

You can find detailed information on our range of services in the service catalog web page:
https://support.industry.siemens.com/cs/sc

**Industry Online Support app**

You will receive optimum support wherever you are with the "Siemens Industry Online Support" app. The app is available for Apple iOS, Android and Windows Phone:
https://support.industry.siemens.com/cs/ww/en/sc/2067

## 4.2 Links and literature

Table 4-1

| No. | Topic |
|-----|-------|
| \1\ | Siemens Industry Online Support<br>https://support.industry.siemens.com |
| \2\ | Link to the article page of the application example<br>https://support.industry.siemens.com/cs/ww/en/view/109759944 |
| \3\ | SITRAIN course "Basics of WinCC Professional".<br>https://support.industry.siemens.com/cs/ww/en/view/109758618 |
| \4\ | SITRAIN: You can find out more about the training and courses as well as their locations and dates at: https://www.siemens.com/sitrain |

## 4.3 Change documentation

Table 4-2

| Version | Date | Change |
|---------|------|--------|
| V1.0 | 10/2018 | First version |