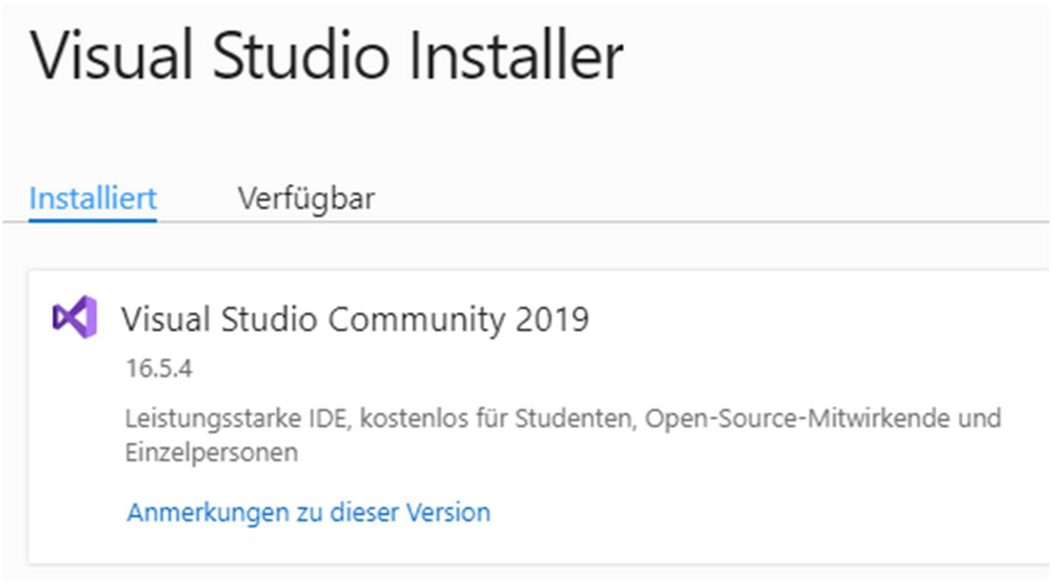
Tutorial

# Windows Forms – GUI programming with C#

## Inhalt

## Installation

First of all, you need to install Visual Studio. As a student you can get the enterprise version for free but in this case the free community version should also be enough.



Visual Studio Installer

Installiert    Verfügbar

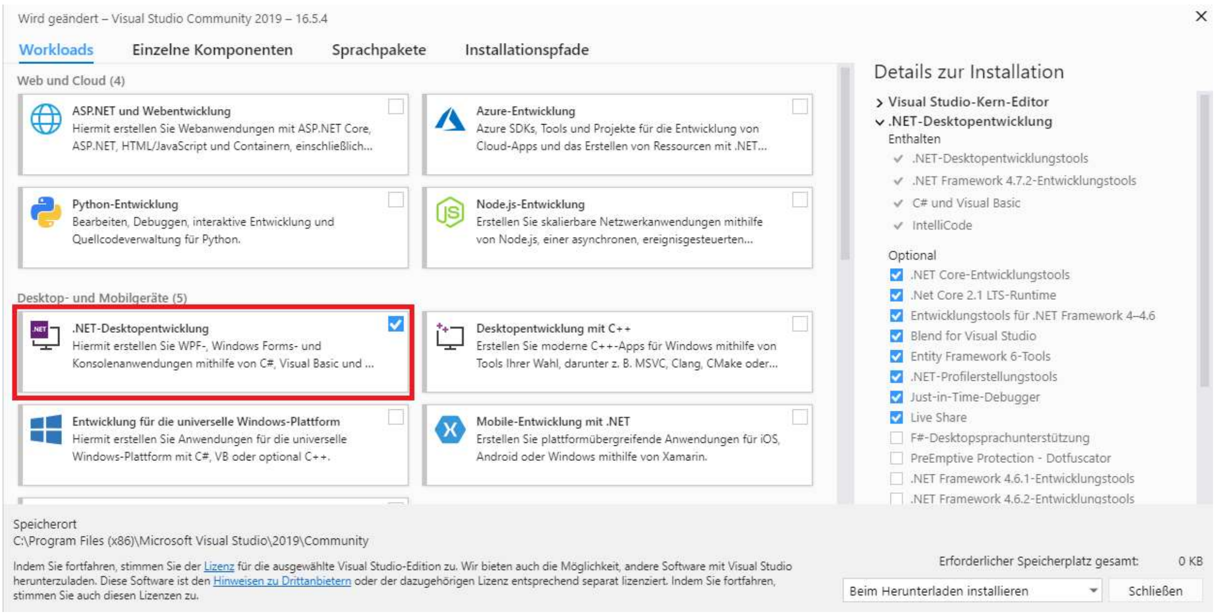
Visual Studio Community 2019

16.5.4

Leistungsstarke IDE, kostenlos für Studenten, Open-Source-Mitwirkende und Einzelpersonen

Anmerkungen zu dieser Version

If done, the required package needs to be installed. To build up a GUI with the Windows Forms Framework, we need the package called **".NET-Desktopentwicklung"**.



Wird geändert – Visual Studio Community 2019 – 16.5.4

Workloads    Einzelne Komponenten    Sprachpakete    Installationspfade

Web und Cloud (4)

- [ ] **ASP.NET und Webentwicklung** – Hiermit erstellen Sie Webanwendungen mit ASP.NET Core, ASP.NET, HTML/JavaScript und Containern, einschließlich...
- [ ] **Azure-Entwicklung** – Azure SDKs, Tools und Projekte für die Entwicklung von Cloud-Apps und das Erstellen von Ressourcen mit .NET...
- [ ] **Python-Entwicklung** – Bearbeiten, Debuggen, interaktive Entwicklung und Quellcodeverwaltung für Python.
- [ ] **Node.js-Entwicklung** – Erstellen Sie skalierbare Netzwerkanwendungen mithilfe von Node.js, einer asynchronen, ereignisgesteuerten...

Desktop- und Mobilgeräte (5)

- [✓] **.NET-Desktopentwicklung** – Hiermit erstellen Sie WPF-, Windows Forms- und Konsolenanwendungen mithilfe von C#, Visual Basic und ...
- [ ] **Desktopentwicklung mit C++** – Erstellen Sie moderne C++-Apps für Windows mithilfe von Tools Ihrer Wahl, darunter z. B. MSVC, Clang, CMake oder...
- [ ] **Entwicklung für die universelle Windows-Plattform** – Hiermit erstellen Sie Anwendungen für die universelle Windows-Plattform mit C#, VB oder optional C++.
- [ ] **Mobile-Entwicklung mit .NET** – Erstellen Sie plattformübergreifende Anwendungen für iOS, Android oder Windows mithilfe von Xamarin.

Details zur Installation

› Visual Studio-Kern-Editor

⌄ .NET-Desktopentwicklung

Enthalten
- ✓ .NET-Desktopentwicklungstools
- ✓ .NET Framework 4.7.2-Entwicklungstools
- ✓ C# und Visual Basic
- ✓ IntelliCode

Optional
- [✓] .NET Core-Entwicklungstools
- [✓] .Net Core 2.1 LTS-Runtime
- [✓] Entwicklungstools für .NET Framework 4–4.6
- [✓] Blend for Visual Studio
- [✓] Entity Framework 6-Tools
- [✓] .NET-Profilerstellungstools
- [✓] Just-in-Time-Debugger
- [✓] Live Share
- [ ] F#-Desktopsprachunterstützung
- [ ] PreEmptive Protection - Dotfuscator
- [ ] .NET Framework 4.6.1-Entwicklungstools
- [ ] .NET Framework 4.6.2-Entwicklungstools

Speicherort
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community

Indem Sie fortfahren, stimmen Sie der Lizenz für die ausgewählte Visual Studio-Edition zu. Wir bieten auch die Möglichkeit, andere Software mit Visual Studio herunterzuladen. Diese Software ist den Hinweisen zu Drittanbietern oder der dazugehörigen Lizenz entsprechend separat lizenziert. Indem Sie fortfahren, stimmen Sie auch diesen Lizenzen zu.

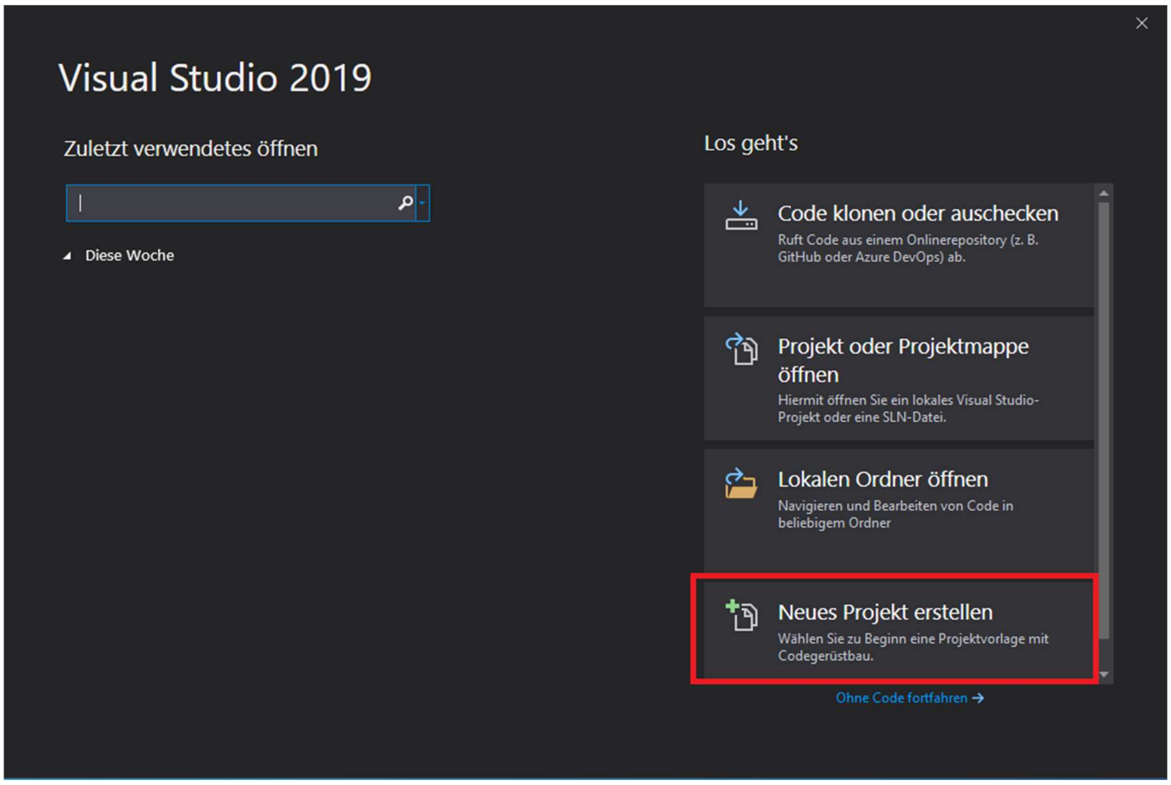
Erforderlicher Speicherplatz gesamt: 0 KB

Beim Herunterladen installieren    Schließen

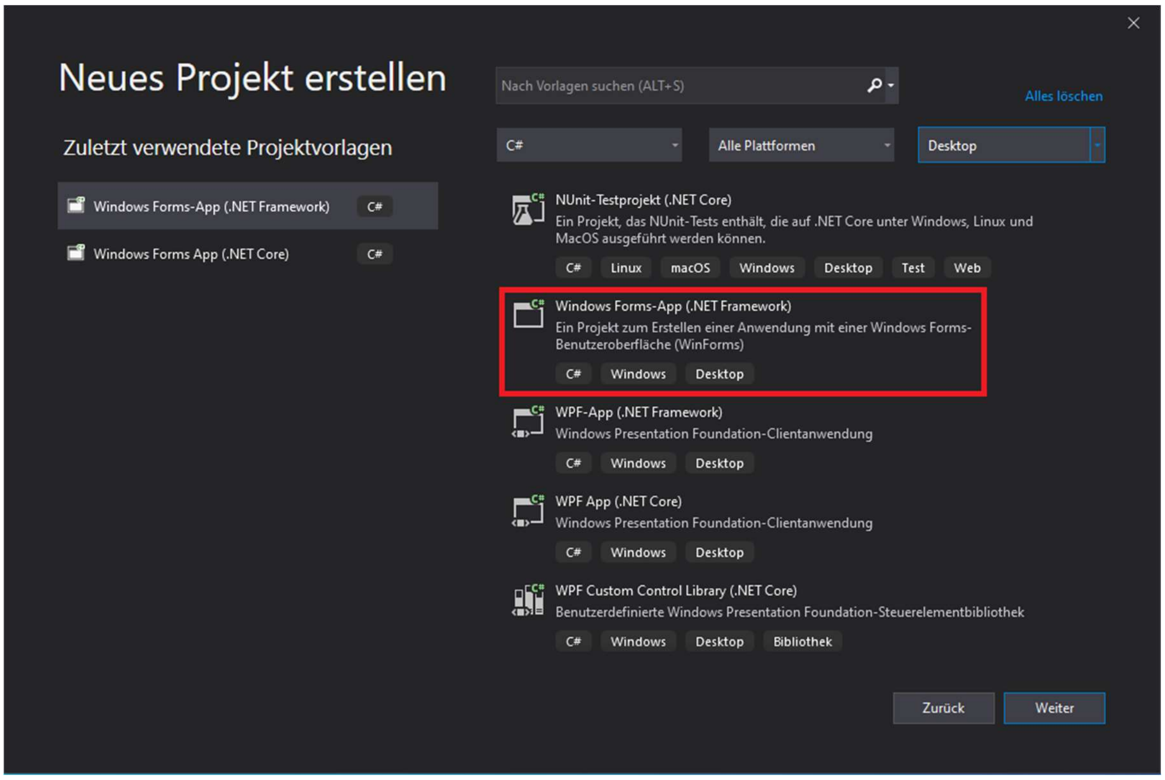
Now you are done with the installing part.

## Create Project

After having installed everything, you can create a project. Start this process by clicking on the marked field called “Neues Projekt erstellen” (→ “**create new project**”).

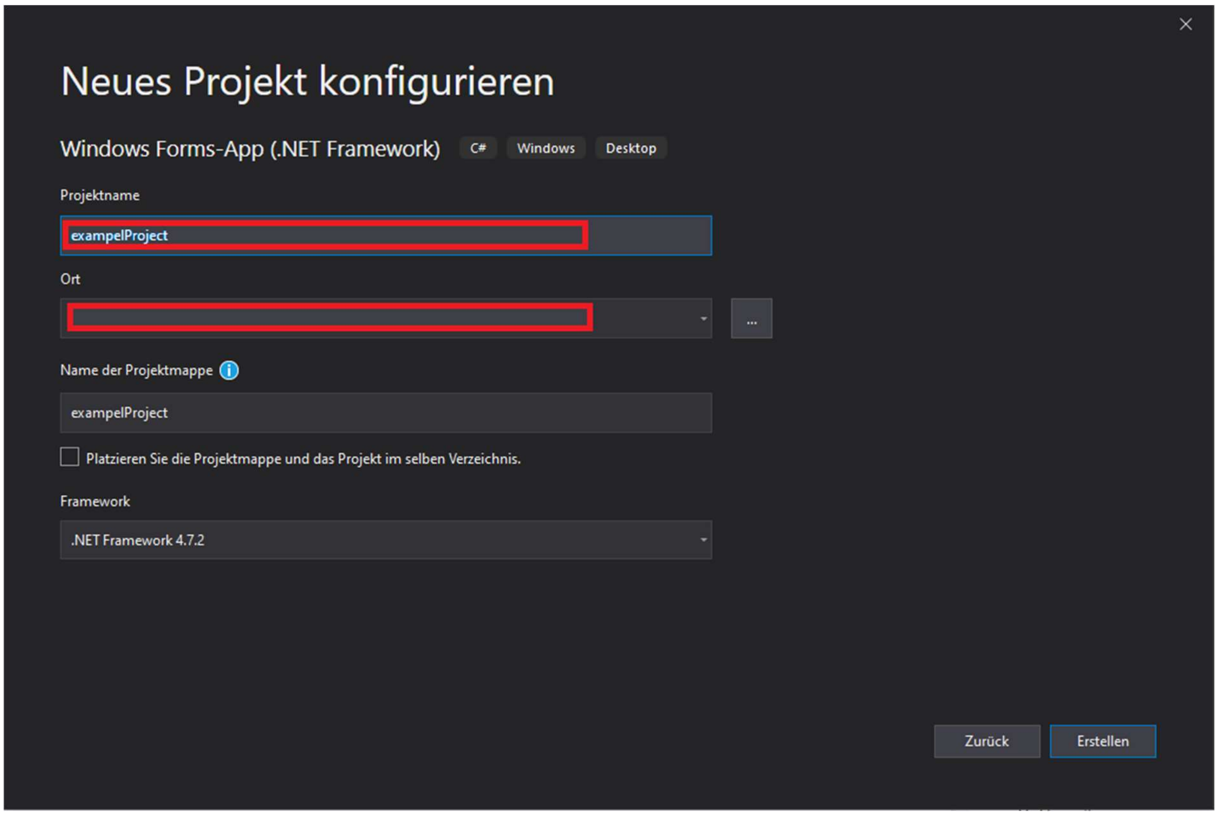

Visual Studio 2019
Zuletzt verwendetes öffnen
Diese Woche
Los geht's
Code klonen oder auschecken
Ruft Code aus einem Onlinerepository (z. B. GitHub oder Azure DevOps) ab.
Projekt oder Projektmappe öffnen
Hiermit öffnen Sie ein lokales Visual Studio-Projekt oder eine SLN-Datei.
Lokalen Ordner öffnen
Navigieren und Bearbeiten von Code in beliebigem Ordner
Neues Projekt erstellen
Wählen Sie zu Beginn eine Projektvorlage mit Codegerüstbau.
Ohne Code fortfahren →

In the following it is important to find the right template. Set “**c#**” and “**desktop**” as a filter and select the marked template where it says “**.NET Framework**” in brackets.


Neues Projekt erstellen
Zuletzt verwendete Projektvorlagen
Windows Forms-App (.NET Framework) C#
Windows Forms App (.NET Core) C#
Nach Vorlagen suchen (ALT+S)
Alles löschen
C#
Alle Plattformen
Desktop
NUnit-Testprojekt (.NET Core)
Ein Projekt, das NUnit-Tests enthält, die auf .NET Core unter Windows, Linux und MacOS ausgeführt werden können.
C# Linux macOS Windows Desktop Test Web
Windows Forms-App (.NET Framework)
Ein Projekt zum Erstellen einer Anwendung mit einer Windows Forms-Benutzeroberfläche (WinForms)
C# Windows Desktop
WPF-App (.NET Framework)
Windows Presentation Foundation-Clientanwendung
C# Windows Desktop
WPF App (.NET Core)
Windows Presentation Foundation-Clientanwendung
C# Windows Desktop
WPF Custom Control Library (.NET Core)
Benutzerdefinierte Windows Presentation Foundation-Steuerelementbibliothek
C# Windows Desktop Bibliothek
Zurück
Weiter

.NET is a software platform from Microsoft with which a so-called .NET application can be executed. In order to be able to run such .NET applications, the so-called .NET Framework is required.

The new project has to have a project name and a location. Think about your own **name** and insert it in the **first text field** and select a **path** in the **second one**.



Neues Projekt konfigurieren

Windows Forms-App (.NET Framework) C# Windows Desktop

Projektname: exampelProject

Ort:

Name der Projektmappe: exampelProject

☐ Platzieren Sie die Projektmappe und das Projekt im selben Verzeichnis.

Framework: .NET Framework 4.7.2

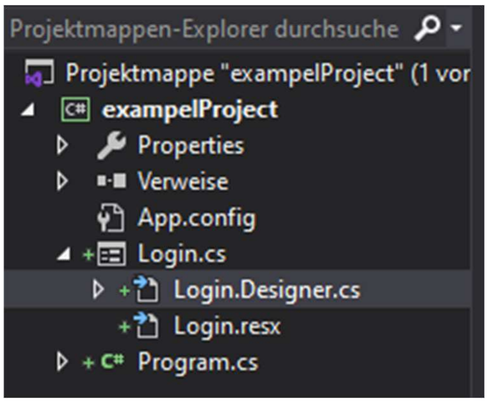
With a click on the button “Erstellen” (→”**create**”), the project will be created. This may take longer time – don`t worry about it.

## Programming

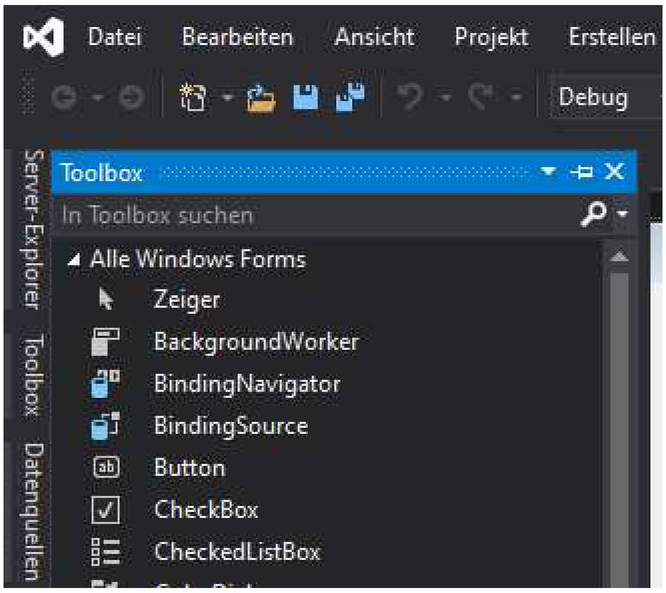
### The IDE

On the right side, you can find your project files. The selected “Login.**Designer.cs”** provides windows generated code and will be treated in the following topic. The “Login.cs” file provides the code behind the main form. In a new project it is called “**Form1.cs**” by default.

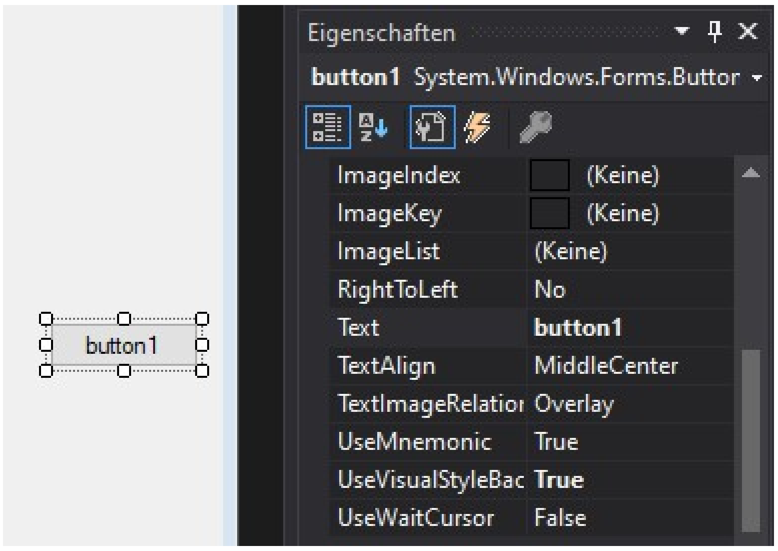
If you cancelled the overview by accident, you can redisplay it by typing the hotkey “Strg+Alt+L” (in German) or you search it in the top tap “**View**”.



On the left side you should see the “**Toolbox**”. The toolbox provides a lot of predefined windows elements which you can add to your form by double-click or drag-and-drop.



If you select an item in your form, which you previously added from the toolbox, you can **edit its properties**. The properties can be found on the right side under your project files.
This is the main method to edit an items name, displayed text, colour and everything else. You could also do this by editing the windows generated code in the designer but I highly advise against doing so because if you produce an error it is difficult to undo and fix it.

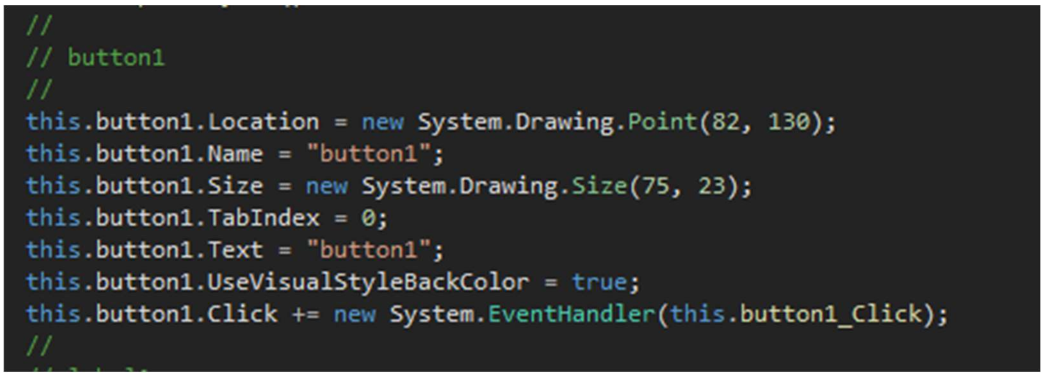


| | |
|---|---|
| ImageIndex | (Keine) |
| ImageKey | (Keine) |
| ImageList | (Keine) |
| RightToLeft | No |
| Text | **button1** |
| TextAlign | MiddleCenter |
| TextImageRelatior | Overlay |
| UseMnemonic | True |
| UseVisualStyleBac | **True** |
| UseWaitCursor | False |

The **flash** beneath the blue marked property-symbol provides predefined **events**. Those events can be activated and edited later on in the code.

## The Designer-Code

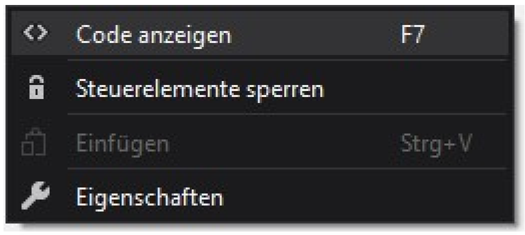
By adding items from the toolbox to your form, windows generates the required code for you. This happens in the **Designer.cs** file. You can find information like names and positions of each element here.



```
//
// button1
//
this.button1.Location = new System.Drawing.Point(82, 130);
this.button1.Name = "button1";
this.button1.Size = new System.Drawing.Size(75, 23);
this.button1.TabIndex = 0;
this.button1.Text = "button1";
this.button1.UseVisualStyleBackColor = true;
this.button1.Click += new System.EventHandler(this.button1_Click);
//
```

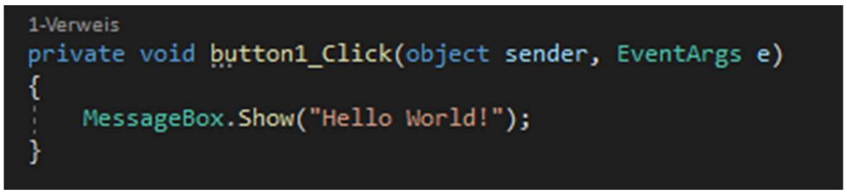
## Your Code

You can get to your own code behind the GUI by right-clicking the main form and then “Code anzeigen” (→”**view code**”) or pressing “F7”.



In this file you can fill operate the exercises of your GUI. Means, you can define your variables, functions and loops.

In this example I defined an event which gets called by **clicking my button** (to define the event, double-click on the button, or take the longer way of defining it in the event properties → **flash**-symbol).



```
1-Verweis
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show("Hello World!");
}
```

Every button-click will open a message-box which displays the text “Hello World!”.

Other possibilities, as well as my example project code will be discussed during the presentation.