Richard Kaiser

www.rkaiser.de

C# Grundkurs

mit

Microsoft Visual Studio 2015/2017

Inhalt

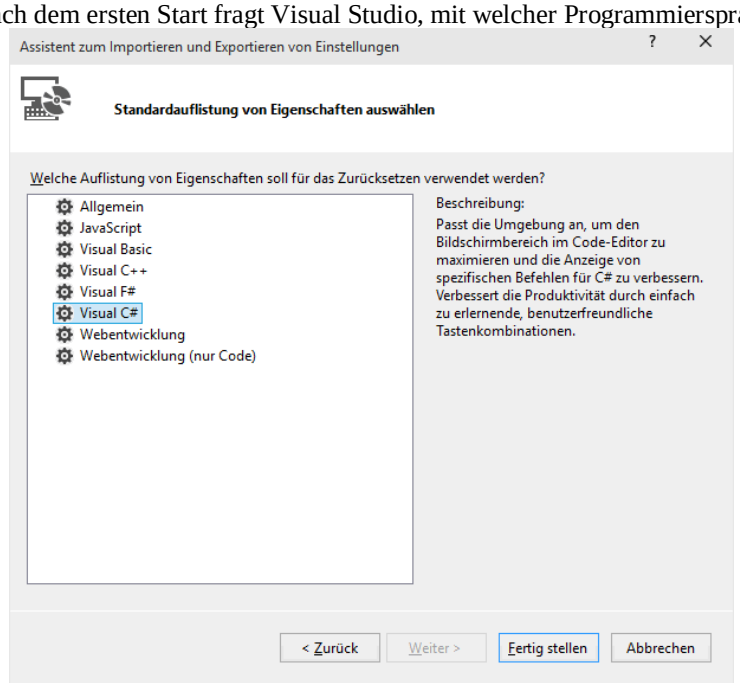
1 Die Entwicklungsumgebung

Visual Studio besteht aus verschiedenen Werkzeugen (Tools), die einen Programmierer bei der Entwicklung von Software unterstützen. Eine solche Zusammenstellung von Werkzeugen zur Softwareentwicklung bezeichnet man auch als Programmier- oder **Entwicklungsumgebung**.

Einfache Entwicklungsumgebungen bestehen nur aus einem Editor und einem Compiler. Für eine effiziente Entwicklung von komplexeren Anwendungen (dazu gehören viele Windows-Anwendungen) sind aber oft weitere Werkzeuge notwendig. Wenn diese wie in Visual Studio in einem einzigen Programm integriert sind, spricht man auch von einer **integrierten Entwicklungsumgebung** (engl.: „integrated development environment“, **IDE)**.

In diesem Kapitel wird zunächst an einfachen Beispielen gezeigt, wie man mit Visual Studio 2015 Windows-Programme mit einer grafischen Benutzeroberfläche entwickeln kann. Anschließend (ab Abschnitt 1.4) werden dann die wichtigsten Werkzeuge von Visual Studio ausführlicher vorgestellt. Für viele einfache Anwendungen (wie z.B. die Übungsaufgaben) reichen die Abschnitte bis 1.8. Die folgenden Abschnitte sind nur für anspruchsvollere oder spezielle Anwendungen notwendig. Sie sind deshalb mit dem Zeichen Θ gekennzeichnet und können übergangen werden. Weitere Elemente der Entwicklungsumgebung werden später beschrieben, wenn sie dann auch eingesetzt werden können.

Nach dem ersten Start fragt Visual Studio, mit welcher Programmiersprache bevorzugt gearbeitet wird:

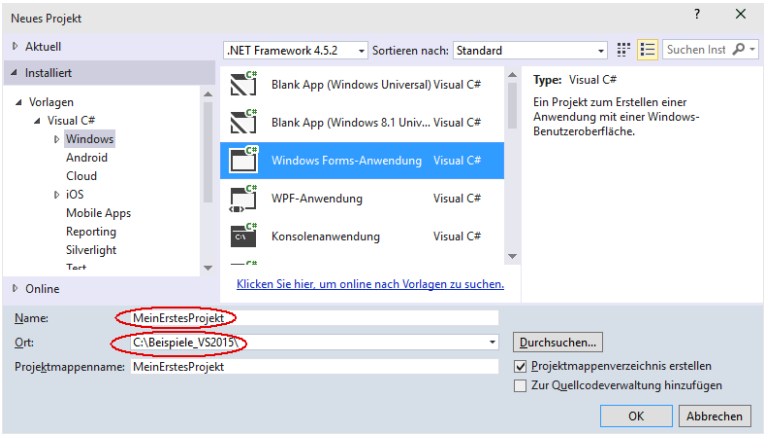


Wenn Sie mit C# arbeiten, wählen Sie hier **Visual C#**. Sie können diese Einstellungen jederzeit mit *Extras*|*Einstellungen importieren und exportieren* ändern.

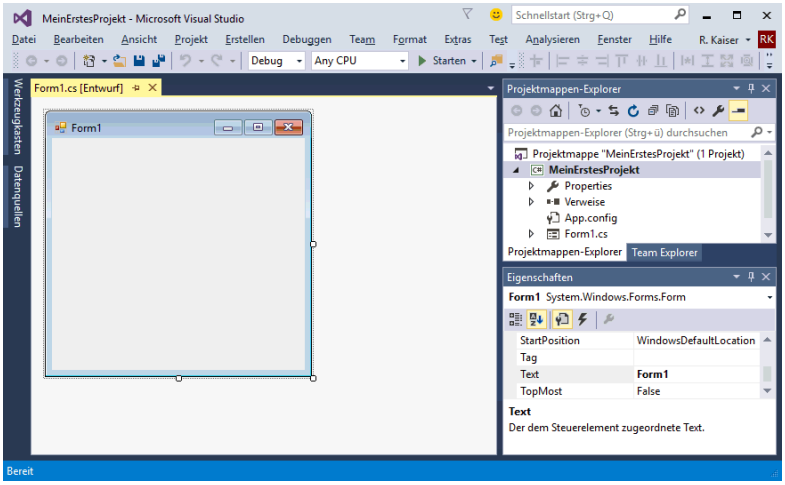
1.1 Visuelle Programmierung: Ein erstes kleines Programm

In Visual Studio 2015 findet man unter ***Datei*|*Neu*|*Projekt*|*Installiert*** Vorlagen für verschiedene Arten von Anwendungen. Eine Windows-Anwendung mit einer graphischen Benutzeroberfläche erhält man mit der Vorlage

Windows Forms-Anwendung. Ein solches Projekt legt man dann an, indem man nach *Name* einen Namen und nach *Ort* ein Verzeichnis für das Projekt eingibt und dann den OK-Button anklickt:

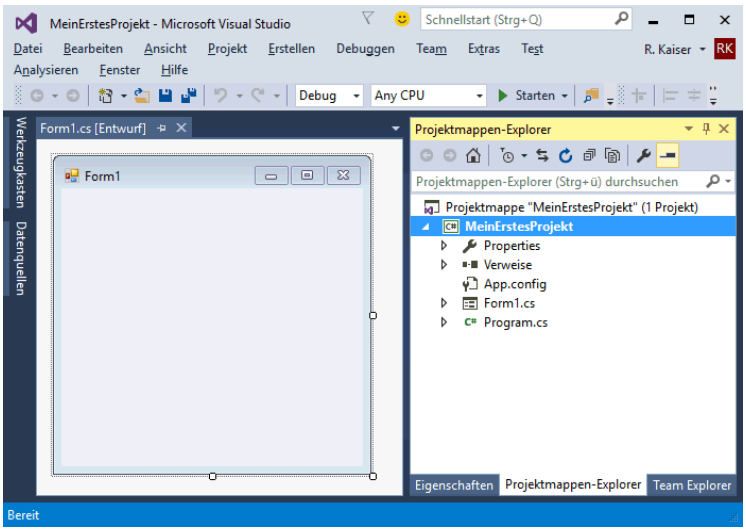


Anschließend sieht Visual Studio etwa so aus:

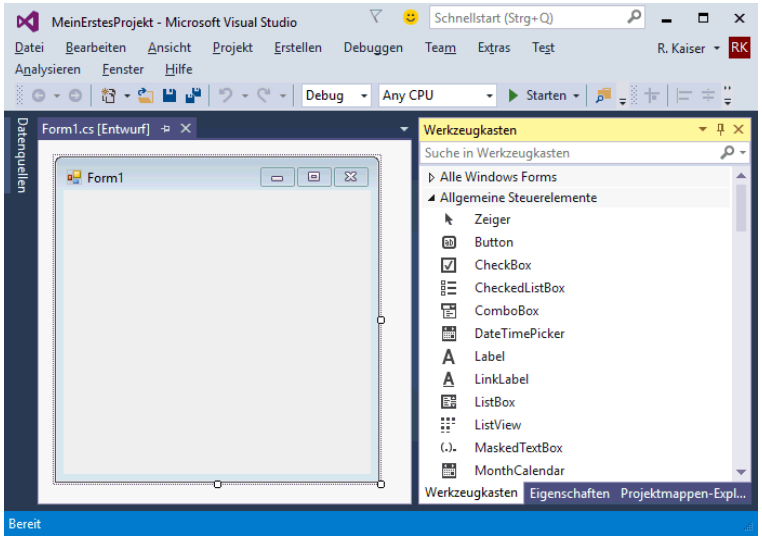


Die Anordnung dieser Elemente hängt von verschiedenen Faktoren ab und kann von dieser Abbildung abweichen.

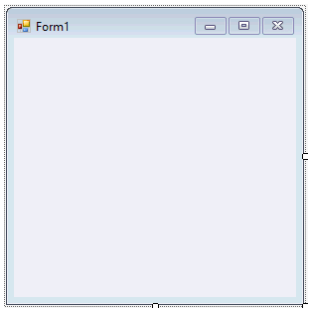
Damit das Eigenschaften-Fenster auf der ganzen Höhe von Visual Studio angezeigt wird, ziehen Sie es auf den Projektmappen-Explorer: Drücken Sie dazu auf der Titelzeile des Eigenschaften-Fensters die linke Maustaste. Fahren Sie dann mit gedrückter Maustaste auf die Titelzeile des Projektmappen-Explorers und lassen Sie dann die Maustaste los. Dann sieht Visual Studio etwa so aus:



Als Nächstes klicken Sie den senkrecht stehenden Schriftzug „Werkzeugkasten“ am linken Rand an und ziehen diesen ebenso auf den Projektmappen-Explorer:

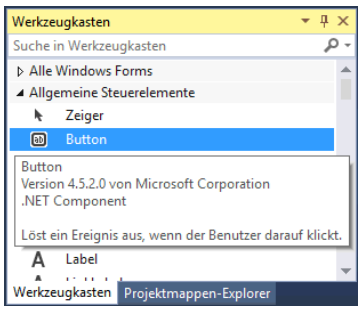


Das **Formular** (hier *Form1*) ist der Ausgangspunkt für alle Windows Forms Anwendungen, die mit Visual Studio entwickelt werden. Es entspricht dem Fenster, das beim Start des Programms angezeigt wird:



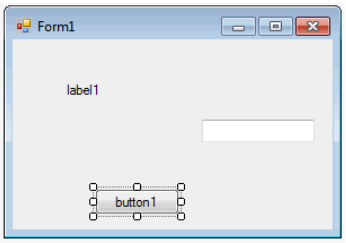
Auf ein Formular kann man **Steuerelemente** (**Controls**) aus dem **Werkzeugkasten** setzen. Der Werkzeugkasten wird nur angezeigt, wenn das Formular angezeigt wird. Er enthält praktisch alle unter Windows üblichen Steuerelemente. Diese sind auf verschiedene Gruppen verteilt (z.B. *Allgemeine Steuerelemente*, *Container* usw.), die auf- und zugeklappt werden können. Die meisten dieser Steuerelemente (wie z.B. ein Button) werden im laufenden Programm auf dem Formular angezeigt.

Falls Ihnen die Namen und die kleinen Icons nicht allzu viel sagen, lassen Sie einfach den Mauszeiger kurz auf einer Zeile des Werkzeugkastens stehen: Dann erscheint ein kleines Hinweisfenster mit einer kurzen Beschreibung.



Um ein **Element** aus dem Werkzeugkasten **auf das Formular** zu **setzen**, zieht man es einfach vom Werkzeugkasten auf das Formular. Oder man klickt mit der Maus zuerst auf die Werkzeugkasten-Zeile (sie wird dann als markiert dargestellt) und dann auf die Stelle im Formular, an die die linke obere Ecke kommen soll.

Beispiel: Nachdem man ein Label (Zeile sieben in *Allgemeine Steuerelemente*, mit dem großen *A*), eine TextBox (vierte Zeile von unten, Aufschrift *ab*) und einen Button (zweite Zeile mit der Aufschrift *ab*) auf das Formular gesetzt hat, sieht es etwa folgendermaßen aus:



Durch diese Spielereien haben Sie **schon ein richtiges Windows-Programm** erstellt – zwar kein besonders nützliches, aber immerhin. Sie können es folgendermaßen starten:

– mit *Debuggen*|*Debugging starten* von der Menüleiste, oder
– mit *F5* von einem beliebigen Fenster in Visual Studio oder
– durch den Aufruf der vom Compiler erzeugten Exe-Datei.

Dieses Programm hat schon viele Eigenschaften, die man von einem Windows-Programm erwartet: Man kann es mit der Maus verschieben, vergrößern, verkleinern und schließen.

Bemerkenswert an diesem Programm ist vor allem der im Vergleich zu einem nichtvisuellen Entwicklungssystem **geringe Aufwand**, mit dem es erstellt wurde. So braucht Petzold in seinem Klassiker „Programmierung unter Windows“ (Petzold 1992, S. 33) ca. 80 Zeilen nichttriviale C-Anweisungen, um den Text „Hello Windows“ wie in einem Label in ein Fenster zu schreiben. Und in jeder dieser 80 Zeilen kann man einiges falsch machen.

Vergessen Sie nicht, Ihr **Programm** zu **beenden**, bevor Sie es weiterbearbeiten. Solange das Programm noch läuft können Sie weder den Compiler erneut starten noch das Formular verändern.

Diese Art der Programmierung bezeichnet man als **visuelle Programmierung**. Während man bei der konventionellen Programmierung ein Programm ausschließlich durch das Schreiben von Anweisungen (Text) in einer Programmiersprache entwickelt, wird es bei der visuellen Programmierung ganz oder teilweise aus vorgefertigten grafischen Komponenten zusammengesetzt.

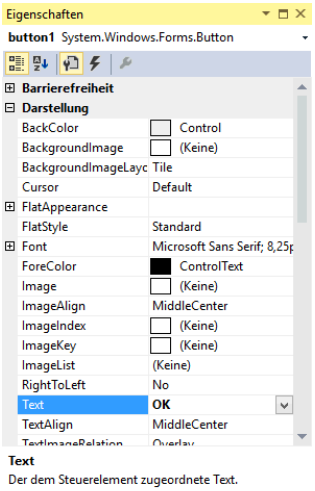
Mit Visual Studio kann die Benutzeroberfläche eines Programms visuell gestaltet werden. Damit sieht man bereits beim Entwurf des Programms, wie es später zur Laufzeit aussehen wird. Die Anweisungen, die als Reaktionen auf Benutzereingaben (Mausklicks usw.) erfolgen sollen, werden dagegen konventionell in einer Programmiersprache (z.B. C#**)** geschrieben.

1.2 Das Eigenschaften-Fenster

Das zuletzt auf einem Formular (bzw. im Pull-down-Menü des Eigenschaftenfensters) angeklickte Steuerelement wird als das **aktuell ausgewählte Steuerelement** bezeichnet. Man erkennt es an den kleinen Quadraten an seinem Rand, den sogenannten **Ziehquadraten**. An ihnen kann man mit der Maus ziehen und so die Größe des Steuerelements verändern. Ein **Formular** wird dadurch zum aktuell ausgewählten Steuerelement, dass man mit der Maus eine freie Stelle im Formular anklickt.

Beispiel: Im letzten Beispiel ist *button1* das aktuell ausgewählte Steuerelement.

Im **Eigenschaften-Fenster** (Kontextmenü des Steuerelements auf dem Formular, oder *Ansicht|Eigenschaftenfenster* – nicht mit *Ansicht|Eigenschaftenseiten* verwechseln) werden die Eigenschaften des aktuell ausgewählten Steuerelements angezeigt. In der linken Spalte stehen die **Namen** und in der rechten die **Werte** der Eigenschaften. Mit der Taste *F1* erhält man eine Beschreibung der Eigenschaft.



Ein Fenster (z.B. das Eigenschaftenfenster) kann man aus Visual Studio lösen, indem man seine Verankerung über das Kontextmenü (rechte Maustaste) der Titelzeile aufhebt:

Dann erhält man ein eigenständiges Fenster wie in der Abbildung rechts oben.

Den Wert einer Eigenschaft kann man über die rechte Spalte verändern. Bei manchen Eigenschaften kann man den neuen Wert über die Tastatur eintippen. Bei anderen wird nach dem Anklicken der rechten Spalte ein kleines Dreieck für ein Pull-down-Menü angezeigt, über das ein Wert ausgewählt werden kann. Oder es wird ein Symbol mit drei Punkten „…“ angezeigt, über das man Werte eingeben kann.

Beispiel: Bei der Eigenschaft ***Text*** kann man mit der Tastatur einen Text eingeben. Bei einem Button ist dieser Text die Aufschrift auf dem Button (z.B. „OK“), und bei einem Formular die Titelzeile (z.B. „Mein erstes C#-Programm“).
Bei der Eigenschaft ***BackColor*** (z.B. bei einem Button) kann man über ein Pull-down-Menü die **Hintergrundfarbe** auswählen.
Klickt man die rechte Spalte der Eigenschaft ***Font*** und danach das Symbol „…“ an, kann man die **Schriftart** der Eigenschaft *Text* auswählen.

Ein Steuerelement auf dem Formular wird nicht nur an seine Eigenschaften im Eigenschaftenfenster angepasst, sondern auch umgekehrt: Wenn man die Größe durch Ziehen an den Ziehquadraten auf dem Formular verändert, werden die Werte der entsprechenden Eigenschaften (*Location* und *Size* im Abschnitt *Layout*) im Eigenschaftenfenster automatisch aktualisiert.

1.3 Erste Schritte in C#

Als nächstes soll das Programm aus dem letzten Abschnitt so erweitert werden, dass als Reaktion auf Benutzereingaben (z.B. beim Anklicken eines Buttons) Anweisungen ausgeführt werden.

Windows-Programme können Benutzereingaben in Form von Mausklicks oder Tastatureingaben entgegennehmen. Im Unterschied zu einfachen Konsolen-Programmen (z.B. DOS-Programmen) muss man in einem Windows-Programm aber keine speziellen Funktionen (wie z.B. *scanf* in C) aufrufen, die auf solche Eingaben warten. Stattdessen werden alle Eingaben von Windows zentral entgegengenommen und als sogenannte Botschaften (Messages) an das entsprechende Programm weitergegeben. Dadurch wird in diesem Programm ein sogenanntes **Ereignis** ausgelöst.

Die Ereignisse, die für das aktuell ausgewählte Steuerelement eintreten können, werden nach dem Anklicken des Symbols für die **Ereignisse** im Eigenschaftenfenster angezeigt.