MASTER CLASS

Dirk Frischalowski

Visual C# 2008

Einstieg für Anspruchsvolle

Visual C# 2008 Express Edition,
Lerntest, Lösungen, Beispiele

DVD

ADDISON-WESLEY [in Kooperation mit] PEARSON Studium

3

Visual C# 2008 Express Edition

3.1 Einführung

3.1.1 Allgemeines

Ein Vorteil des .NET Frameworks ist es, dass Sie es kostenfrei aus dem Internet laden und verwenden können. Allerdings ist die Eingabe des Sourcecodes mit einem normalen Text-Editor, das Übersetzen und Debuggen auf Kommandozeilenebene und überhaupt die Arbeit mit den Kommandozeilentools nicht sonderlich effektiv. Für die professionelle Entwicklung sollte daher auf eine IDE (Integrated Development Environment – Integrierte Entwicklungsumgebung) zurückgegriffen werden. Mit der C# Express Edition (und denen für Visual Basic und C++) stellt Microsoft eine kostenfreie IDE zur Verfügung (*http://www.microsoft.com/germany/express/*), die der »professionellen« Version durchaus das Wasser reichen kann.

Tipp

Mit SharpDevelop (kurz #develop oder #D), welches Sie unter *http://www.icsharpcode.net/* beziehen können, existiert eine weitere kostenfreie IDE zur Entwicklung mit dem .NET Framework. Aktuell befindet sich die IDE noch im Beta-Status.

3.1.2 Versionitis

Es gibt jeweils eine Visual Studio Express Edition zu C#, Visual Basic und C++. Außerdem gibt es eine Edition zur Entwicklung von Webanwendungen (Visual Web Developer) und eine Express Edition des SQL Servers 2005. Alle Produkte sind dauerhaft kostenfrei nutzbar, auch für kommerzielle Anwendungen.

Diese kleinen Versionen des Visual Studios bieten die Standardmerkmale des Visual Studios (Übersetzen und Debuggen von Anwendungen, Projektverwaltung, Konfiguration, leistungsfähiger Editor usw.). Weiterhin gibt es noch eine Standard-, Professional- und Team System-Version des Visual Studio 2008. Diese unterscheiden sich in so vielen Details, dass ein Blick in die Tabellen des Produktvergleichs hilfreich ist (aktuell auf der Site *http://msdn2.microsoft.com/de-de/vs2008/cc149003.aspx*).

Die großen Versionen des Visual Studios beinhalten z. B. alle Programmiersprachen, also C#, Visual Basic und C++ (kein J# mehr). Außerdem kommt noch die Webentwicklung hinzu. Die Team Systems-Versionen beinhalten z. B. leistungsfähige Datenbank- und Codeanalyse-Tools. Die Express Edition verfügt aber bereits über eine Integration zu lokalen Datenbanken mit dem SQL Server Compact Edition.

Hinweis

Im Folgenden wird die Installation und Verwendung von Visual C# 2008 Express erläutert und als Basis für die weitere Arbeit mit diesem Buch genutzt. Die meisten Informationen sind allerdings unabhängig von der Verwendung dieser Version.

3.1.3 Installation

Zur Installation der Visual C# 2008 Express Edition stehen mehrere Varianten zur Verfügung. Die für Sie günstigste Vorgehensweise ist die Installation über die beiliegende DVD zum Buch. Die Vorgehensweise entspricht weitestgehend den beiden Varianten, die nachfolgend vorgestellt werden.

Bevor Sie beginnen, stellen Sie noch sicher, dass sich keine veralteten Versionen des .NET Frameworks 3.5 bzw. des Visual Studio 2008 auf Ihrem Rechner befinden, da die Installation sonst nicht erfolgreich durchgeführt werden kann. Mit vorhandenen Versionen des Visual Studios 2003 oder 2005 verträgt sich die Installation problemlos.

Webinstallation

Über die Webinstallation werden nur die ausgewählten Features der Express-Version installiert. Nachteilig daran ist, dass Sie natürlich eine Internetverbindung benötigen und dass Sie bei einer Neuinstallation die Daten wiederum laden müssen. Unter dem URL *http://www.microsoft.com/germany/express/download/default.aspx* finden Sie die Webinstallationen der verschiedenen Express-Versionen. Für C# laden Sie über den Link *Download* in dem grünen Symbol die Datei *vcssetup.exe* und starten sie danach. Nach der Auswahl der zu installierenden Komponenten werden diese in der aktuellen Version über das Web geladen. Das kann aber eine ganze Weile dauern. Neben dem .NET Framework 3.5 und der Express Edition von C# können Sie optional das MSDN (Microsoft Developer Network – Hilfe zum Visual Studio und C#) und die Silverlight Runtime (ein Browser-Addin für die Anzeige von WPF-Anwen-

dungen) installieren. Dies wären zusätzlich 311 Mbyte für das MSDN und 1.4 Mbyte für Silverlight. Wenn Sie nur die Visual C# 2008 Express Edition installieren wollen, bleiben nur noch ca. 64 Mbyte übrig.

Offline- und DVD-Installation

Die Offline- bzw. DVD-Version (eine DVD kann bestellt werden) enthält sämtliche Daten, die Sie zur Installation aller Express-Versionen benötigen, also auch die Installationsdaten für Visual Basic, C++ und die Webentwicklung. Dies ist beispielsweise für die häufige Installation der Express Versionen auf unterschiedlichen Rechnern nützlich. Microsoft stellt auch diese Version unter dem URL *http://www.microsoft.com/germany/express/download/default.aspx* zur Verfügung. Zu laden ist eine ca. 3 Gbyte große ISO-Datei, d. h. ein DVD-Abbild. Dieses Abbild lässt sich mit DVD-Brennprogrammen wie Nero verwenden, um eine Installations-DVD zu erzeugen. Bei genügend Festplattenplatz entpacken Sie die ISO-Datei einfach mit dem Programm WinRAR (*http://www.winrar.de/*) in ein beliebiges Verzeichnis und führen die Installation über dieses Verzeichnis aus. Alternativ verwenden Sie ein Tool wie Daemon, um das Abbild als weiteres Laufwerk bereitzustellen (*http://www.daemon-tools.cc/*).

Nach dem Entpacken bzw. Brennen der ISO-Datei steht im Wurzelverzeichnis die Datei *setup.hta* zur Verfügung. Diese wird in einem Browserfenster geöffnet und bietet über ein grafisches Menü die Installation der gewünschten Express Edition an.

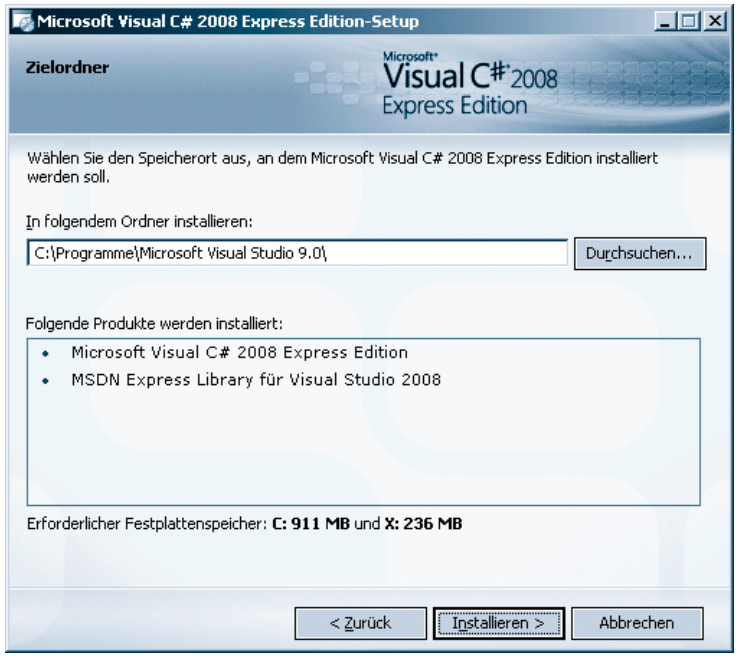
1. Wählen Sie beispielsweise die Installation von C# aus. Nach der Anzeige des Willkommen-Dialogs bestätigen Sie die Schaltfläche WEITER.
2. Markieren Sie im folgenden Dialog die Option, um den Lizenzvertrag anzunehmen, und bestätigen Sie mit WEITER.
3. Markieren Sie in den Installationsoptionen, dass Sie die MSDN Express Library für Visual Studio 2008 installieren möchten. Dadurch wird die komplette Dokumentation zum Visual Studio, zu C# und der Arbeit mit dem .NET Framework installiert. Dies macht auch den größten Teil der Installation aus.
4. Weitere Einstellmöglichkeiten gibt es nicht. Klicken Sie auf WEITER. Es werden noch einmal alle Installationsbestandteile angezeigt. Nach dem Klick auf INSTALLIEREN, wird sofort mit der Installation begonnen. Diese benötigt ca. 1,2 Gbyte Festplattenplatz, siehe Abbildung 3.1.

   Als einzige weitere Option können Sie noch das Installationsverzeichnis festlegen. Wenn nicht anders angegeben, wird im Buch immer von der Standardinstallation in das Verzeichnis *C:\Programme\Microsoft Visual Studio 9* ausgegangen.
5. Nach der erfolgreichen Installation wird ein letzter Dialog angezeigt, den Sie über BEENDEN schließen. Ein Reboot ist normalerweise während und nach der Installation nicht notwendig.

Hinweis

Die Installation des Silverlight-Addins wird in der Offline-Installation nicht mit angeboten.

Abbildung 3.1
Bestandteile der Visual C# 2008 Express Edition



Um die Express Edition von C# zu starten, gehen Sie über das START-Menü und den Ordner *Programme* und klicken auf den Eintrag *Microsoft Visual C# 2008 Express Edition*.

Registrierung

Eine Registrierung für das Produkt ist bei einer Offline-Installation nicht erforderlich. Wenn Sie allerdings die Webinstallation durchgeführt haben, wird auf eine Registrierung hingewiesen. Diese berechtigt Sie zum Zugriff auf weitere Ressourcen, ist aber für die Nutzung des Produkts nicht notwendig.

3.2 Der erste Start

3.2.1 Die Startseite

Nach dem ersten Start erhalten Sie eine wenig spektakuläre Ansicht, welche nur die Startseite und rechts den Projektmappen-Explorer zeigt. Über die Startseite können Sie bereits erstellte Projekte öffnen und im Bereich ERSTE SCHRITTE Bekanntschaft mit der Hilfe schließen.

- Klicken Sie im Bereich ERSTE SCHRITTE auf den Eintrag EINE ERSTE ANWENDUNG ERSTELLEN, um die Hilfe zu öffnen und zu konfigurieren.
- Beim ersten Aufruf der Hilfe wird ein Dialogfenster geöffnet, über das Sie den Zugriff darauf einstellen (Abbildung 3.2). Idealerweise sollte die lokale Hilfe verwendet werden, damit nicht jedes Mal eine Internetverbindung aufgebaut wird. Die lokale Hilfe ist sicher nicht immer die aktuellste, aber erstens können Sie über den Menüpunkt HILFE/AUF AKTUALISIERUNGEN ÜBERPRÜFEN Ihre Installation jederzeit updaten und außerdem gibt es ja auch noch Google. Wenn Sie die Option ONLINEHILFE NICHT VERWENDEN markieren, wird nur in der lokalen Hilfe gesucht.

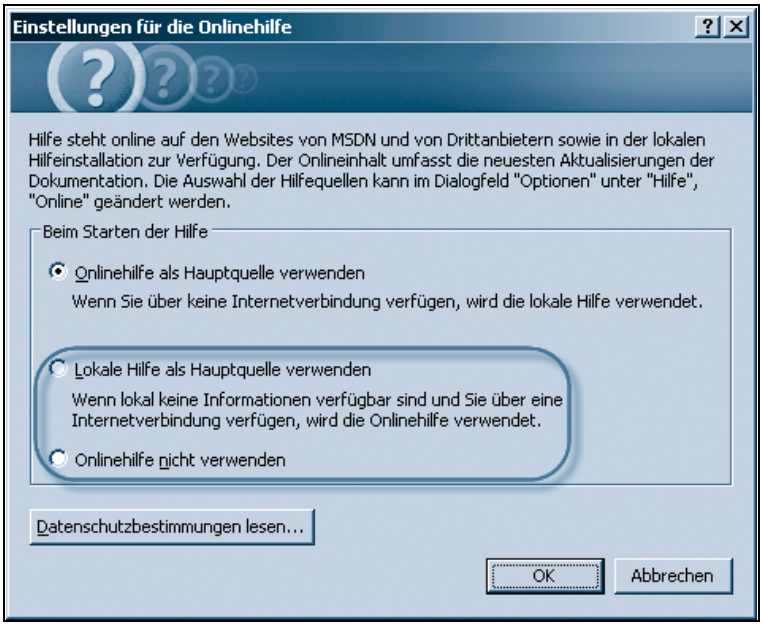


Abbildung 3.2
Konfiguration der Hilfe

- Bestätigen Sie Ihre Einstellungen mit OK.
- Jetzt wird das gewünschte Hilfethema in einem separaten Fenster geöffnet. Es erläutert die Erstellung einer Anwendung, die einen rudimentären Webbrowser darstellt. Das Beispiel ist als Einstieg recht umfangreich, zeigt aber viele Möglichkeiten der Anwendungsentwicklung mit dem Visual Studio.

3.2.2 Eine Konsolenanwendung erstellen

Eine Anwendung für die Konsole verfügt nicht über die grafischen Möglichkeiten einer Windows-Anwendung. Die Ein- und Ausgabe der Daten erfolgt nur im Textmodus. Wozu sind dann solche Anwendungen notwendig bzw. sinnvoll? Zum Beispiel dann, wenn keine grafische Ausgabe notwendig ist und die Systemressourcen geschont werden sollen. Dies ist bei Serveranwendungen häufig der Fall. Ein Client verbindet sich über ein Netzwerk (z. B. das

Internet) mit einem anderen Rechner und nimmt die Dienste eines Servers in Anspruch. Ein einfacher Mail-Server könnte z. B. als Konsolenanwendung realisiert werden.

Weiterhin sind Konsolenanwendungen zum Erlernen der Grundkenntnisse einer Programmiersprache hilfreich, da Sie sich nicht sofort mit zu vielen unbekannten Dingen beschäftigen müssen. Grafische Anwendungen sind zwar attraktiver, benötigen aber etwas mehr Vorkenntnisse.

Hinweis

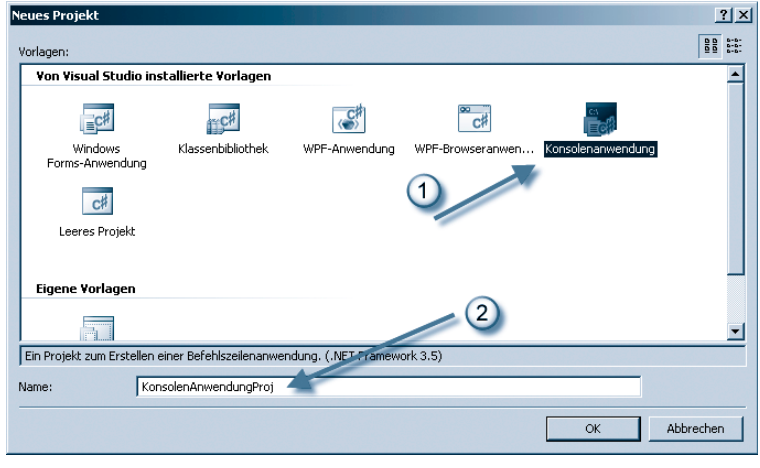
Die folgende Anwendung wird mit der Express Edition von C# erstellt. Die Vorgehensweise kann sich in anderen Versionen leicht unterscheiden.

Beispiel
Eine einfache Konsolenanwendung erstellen

Im Folgenden wird eine einfache Konsolenanwendung erstellt, die einen Text ausgibt und nach dem Betätigen von Enter die Anwendung beendet.

1. Starten Sie das Visual Studio, wenn es nicht bereits geöffnet ist.
2. Klicken Sie auf den Menüpunkt DATEI/NEUES PROJEKT. Es wird das Dialogfenster NEUES PROJEKT geöffnet.
3. Markieren Sie die Projektvorlage KONSOLENANWENDUNG und geben Sie als Projektnamen *KonsolenAnwendungProj* ein (Abbildung 3.3). Bestätigen Sie mit OK. Die Projektnamen enden im Buch immer mit dem Suffix *Proj*.

Abbildung 3.3
Auswahl einer Projektvorlage und Angabe des Projektnamens



Es wird neues Projekt erstellt, das Projekt im Projektmappen-Explorer geladen und der Editor mit der Datei *Program.cs* geöffnet.

4. Markieren Sie die Datei *Program.cs* im Projektmappen-Explorer und benennen Sie die Datei nach *KonsolenAnwendung.cs* um. Die Dateinamen des Hauptprogramms erhalten im Buch in der Regel den Namen des Projekts, allerdings ohne den Suffix *Proj*. Die Frage, ob alle Verweise auf das Codeelement »Program« ebenfalls umbenannt werden sollen beantworten Sie mit JA.

5. Geben Sie in der Methode `Main()` die beiden farbig dargestellten Zeilen ein.

Listing 3.1
Konsolen-Anwendung.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace CSharpBuch.Kap03
{
  class Program
  {
    static void Main(string[] args)
    {
      Console.WriteLine("Halli Hallo");
      Console.ReadLine();
    }
  }
}
```

Um die Anwendung zu übersetzen, haben Sie mehrere Möglichkeiten.

6. Drücken Sie auf die Taste F6, um die Projektmappe inklusive aller darin befindlichen Projekte zu übersetzen. Alternativ können Sie auch den Menüpunkt ERSTELLEN/PROJEKTMAPPE ERSTELLEN auswählen.

Die Anwendung kann nun mit oder ohne Debug-Funktionalität gestartet werden. Dies beeinflusst auch die Ausführung der Anwendung.

7. Drücken Sie auf die Taste F5, um die Anwendung im Debug-Modus auszuführen. In diesem Fall wird ein neues Konsolenfenster geöffnet und die Textausgabe erfolgt darin. Drücken Sie die Taste Enter wird die Anwendung erwartungsgemäß beendet.

   Wenn Sie die Tastenkombination Strg F5 drücken, wird die Anwendung normal (ohne Debugger-Unterstützung) gestartet. Die Ausführung erfolgt schneller und es wird wieder das Konsolenfenster geöffnet. Wenn Sie nun die Enter-Taste drücken, wird die Anwendung nicht sofort beendet, sondern der Text *Drücken Sie eine beliebige Taste ...* angezeigt. Dadurch haben Sie auch nach der Beendigung einer Anwendung die Möglichkeit, deren Ausgabe zu analysieren.

Abbildung 3.4
Start einer Konsolenanwendung ohne Debugger-Unterstützung

Nach dem Beenden der Anwendung befinden Sie sich wieder im Editor und können Ihre Anwendung weiter bearbeiten.

Hinweis

Haben Sie Änderungen an einer Anwendung vorgenommen und starten diese über F5 oder Strg F5, wird die Anwendung vorher automatisch aktualisiert. Sie müssen die Anwendung also nicht manuell vor dem Start aktualisieren, z. B. über F6.

3.2.3 Grundfunktionalität des Visual Studios

Die Anzeige im Visual Studio, d. h. welche Fenster angezeigt werden, hängt immer vom Kontext und Ihren Einstellungen ab. In Abbildung 3.5 sehen Sie das Visual Studio, wie es bei Ihnen nach der Erstellung der Konsolenanwendung aussehen könnte. In (1) können Sie nach Textstellen in der im Editor geöffneten Datei suchen. Der Editor (2) zeigt eine farbige Darstellung des Sourcecodes. Die Liste (3) enthält alle Typen (Klassen, Interfaces...) der geöffneten Datei und erlaubt einen schnellen Wechsel zur Stelle im Editor, an der der Typ definiert wird. Eine zweite Liste (4) zeigt alle Member (Methoden, Instanzvariablen) des ausgewählten Typs an und verzweigt ebenfalls nach Auswahl eines Eintrags an die entsprechende Stelle im Editor.

Die Abbildung zeigt weiterhin einen Fehler in der Anwendung. Statt der Methode `ReadLine()` wurde nur der Text `ReadLin()` geschrieben. Wird eine Anwendung mit einem solchen Fehler übersetzt, öffnet das Visual Studio ein weiteres Fenster (5), die Fehlerliste. Darin wird eine Beschreibung des gefundenen Fehlers sowie dessen Position angezeigt.

Achtung

Wenn sich in einer Anwendung mehrere Fehler befinden, kann es sein, dass weder die Fehlerbeschreibung noch der Fundort den wahren Grund des Fehlers zeigen. Allerdings dient die Anzeige sehr gut als erste Orientierung bei der Fehlersuche.

Schließlich zeigt (6) den aufgeklappten Projektmappen-Explorer.

Wenn Sie ein neues Projekt erstellen, werden vom Visual Studio immer bestimmte Standardeinstellungen vorgenommen. So werden in den Sourcecode standardmäßig die Namespaces `System`, `System.Collections.Generic`, `System.Linq` und `System.Text` eingebunden. Sie können die letzten drei Namespaces meist entfernen, da sie nicht immer benötigt werden. Es ist in diesem Fall günstiger sie später bei Bedarf lieber erneut einzubinden.

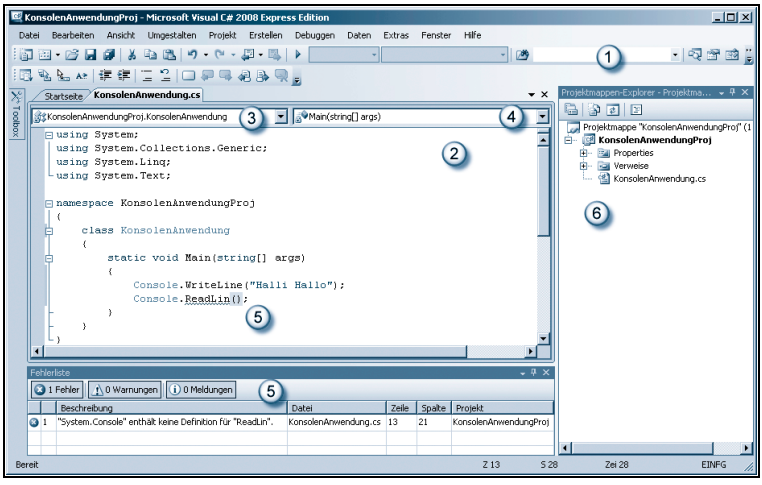


Abbildung 3.5
Visual Studio mit geöffnetem Editor und Fehlerliste

Des Weiteren werden im Ordner *Verweise* im Projekt *KonsolenAnwendungProj* im Projektmappen-Explorer (6) Verweise auf die Assemblies *System.Core* (verschiedene Erweiterungen), *System.Data* und *System.Data.DataSetExtensions* (für den Datenbankzugriff), *System.Xml* (für die Verwendung von XML-Funktionalität) und *System.Xml.Linq* (für die Abfrage von XML-Daten mit LINQ) eingefügt. Auch diese sollten bzw. können Sie entfernen, wenn Sie sie nicht benötigen.

Die Datei *AssemblyInfo.cs* im Knoten *Properties* enthält unter anderem Versionsinformationen und allgemeine Angaben zu Ihrer Anwendung. Die Datei ist nicht zwingend für eine C#-Anwendung erforderlich, sollte aber aus eben genannten Gründen beibehalten werden. Im Kapitel zu Assemblies werden diese Einstellungen näher erläutert.

Tipp

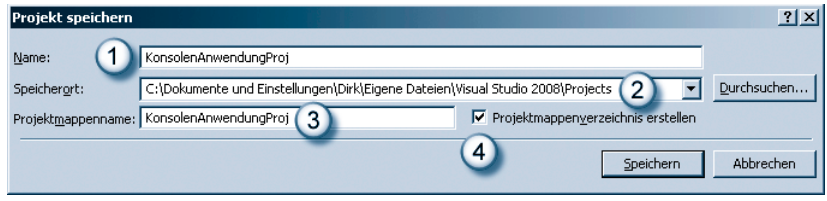
Auch wenn die eingebundenen Namespaces und die Verweise weder die Ausführung noch wesentlich die Anwendungsgröße (bis auf ein paar Byte) beeinflussen, sollten Sie überflüssigen Code und nicht benötigte Einstellungen vermeiden.

3.2.4 Projektverwaltung

Ein Projekt verwaltet sämtliche Einstellungen zu einem bestimmten Anwendungstyp, z. B. einer Konsolenanwendung. Des Weiteren ist ein Projekt immer in einer Projektmappe eingebettet. Wenn Sie also ein neues Projekt erstellen, wird automatisch auch eine Projektmappe erzeugt. Eine Projektmappe kann mehrere Projekte beinhalten, die z. B. zur Lösung einer Aufgabe benötigt werden.

Nachdem Sie das Beispiel *KonsolenAnwendungProj* erstellt haben, ist das Projekt noch nicht gespeichert. Um das Projekt zu speichern, wählen Sie den Menüpunkt DATEI/ALLES SPEICHERN. Es wird ein Dialog angezeigt, in dem Sie den Projektnamen und den Speicherort angeben können. Unter (1) können Sie in der Abbildung 3.6 den Projektnamen festlegen (bzw. beibehalten). Unter (2) wird der Speicherort für Projekt und Projektmappe definiert. Unter (3) wird der Projektmappenname festgelegt, der standardmäßig den Namen des zuerst angelegten Projekts besitzt. Und unter (4) können Sie angeben, ob für die Projektmappe ein weiteres Unterverzeichnis im Speicherort von (3) erstellt werden soll.

Abbildung 3.6
Projekte speichern



Es ergibt sich entsprechend den Einstellungen aus Abbildung 3.6 die folgende Verzeichnisstruktur, die standardmäßig im Verzeichnis *..\Eigene Dateien\Visual Studio 2008\Projects* angelegt wird:

\KonsolenAnwendungProj\KonsolenAnwendungProj.sln
\KonsolenAnwendungProj\KonsolenAnwendungProj.suo
\KonsolenAnwendungProj\KonsolenAnwendungProj
KonsolenAnwendungProj.csproj
\KonsolenAnwendungProj\KonsolenAnwendungProj\KonsolenAnwendung.cs
\KonsolenAnwendungProj\KonsolenAnwendungProj\bin\Debug
\KonsolenAnwendungProj\KonsolenAnwendungProj\bin\Release
\KonsolenAnwendungProj\KonsolenAnwendungProj\obj\...
\KonsolenAnwendungProj\KonsolenAnwendungProj\Properties
AssemblyInfo.cs

Die beiden ersten Dateien beschreiben den Inhalt und die Konfiguration einer Projektmappe (**.sln* – Solution (Projektmappe), **.suo* – Solution User Options (Benutzereinstellungen in einer Projektmappe)). Die beiden anderen Dateien stehen für das Projekt (**.csproj* – C#-Projekt) und den Sourcecode (*Program.cs*). Das Unterverzeichnis *..\Properties* enthält eine weitere Datei *AssemblyInfo.cs*, welche Informationen zur Anwendung enthält, z. B. Copyrightinformationen. Die beiden Unterverzeichnisse *..\bin* und *..\obj* können Sie jederzeit bedenkenlos löschen, da diese beim Erstellen eines Projekts erneut erzeugt werden. Sie enthalten außerdem noch die Unterverzeichnisse *..\Debug* und *..\Release* (Auslieferung), in denen jeweils die Debug- und die Release-Version eines Projekts abgelegt wird.

Wenn Sie ein Projekt oder eine Projektmappe unter einem anderen Namen oder in einem anderen Verzeichnis speichern wollen,

1. Markieren Sie zuerst das Projekt oder die Projektmappe im Projektmappen-Explorer.
2. Wählen Sie den Menüpunkt DATEI/[PROJEKTNAME] SPEICHERN UNTER bzw. DATEI/[PROJEKTMAPPENNAME] SPEICHERN UNTER. Je nach ausgewähltem Eintrag im Projektmappen-Explorer ist immer nur einer dieser Menüpunkte verfügbar.

3.2.5 Das Visual Studio konfigurieren

Wer gibt sich schon mit dem zufrieden, was ihm standardmäßig angeboten wird? Wahrscheinlich niemand. Aus diesem Grund können Sie auch im Visual Studio zahlreiche Dinge konfigurieren, damit sie optimal an Ihre Arbeitsweise angepasst sind, sei es die Farbgebung im Editor, die Texteinrückung oder die angezeigten Symbole der Toolbars. Um das Erscheinungsbild anzupassen:

- Wählen Sie den Menüpunkt EXTRAS/ANPASSEN, um die Symbolleisten und Tastaturkürzel zu bearbeiten.
- Wählen Sie den Menüpunkt EXTRAS/OPTIONEN, um weitere Einstellungen vorzunehmen. Es wird das Dialogfenster OPTIONEN geöffnet (Abbildung 3.7).
- Im Bereich (1) können Sie eine Kategorie bzw. eine Unterkategorie auswählen. Die zugehörigen Einstellungen werden dann im rechten Bereich (2) angezeigt und können dort bearbeitet werden.
- Standardmäßig werden nicht alle möglichen Einstellungen angezeigt. Markieren Sie die Option ALLE EINSTELLUNGEN ANZEIGEN bei (3), damit alle Kategorien eingeblendet werden.

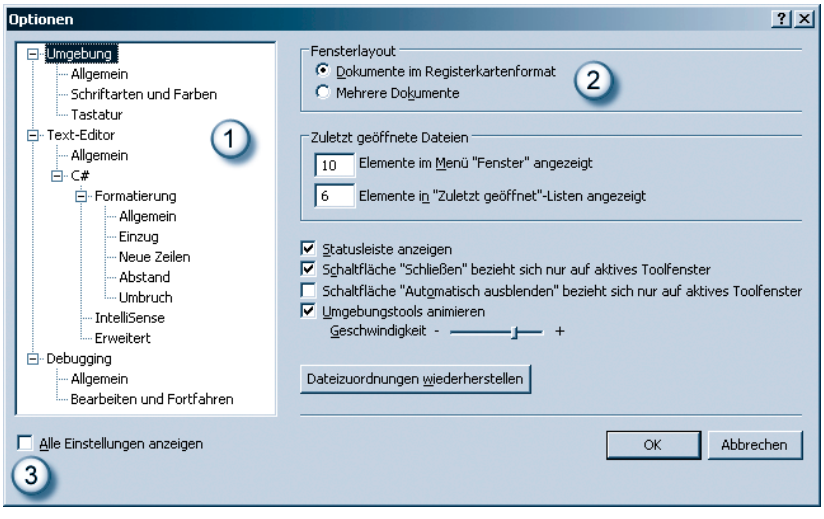


Abbildung 3.7
Die Einstellungen im Visual Studio konfigurieren

Die folgende Tabelle zeigt Ihnen die Position einiger ausgewählter Einstellungen. Wenn Sie einige Zeit mit dem Visual Studio gearbeitet haben, empfiehlt es sich, die gesamten Einstellungsmöglichkeiten noch einmal zu prüfen.

Tabelle 3.1 Ausgewählte Konfigurationsmöglichkeiten im Visual Studio

Kategorie / Unterkategorie	Einstellung / Beschreibung
PROJEKTE UND PROJEKTMAPPEN	Hier legen Sie unter anderem die Standardverzeichnisse zum Speichern von Projekten fest.
UMGEBUNG/HILFE (/ONLINE)	Stellen Sie ein, ob beim Öffnen der Hilfe nur lokal oder auch online gesucht werden soll. Außerdem können Sie angeben, ob Sie die Hilfe lieber direkt im Visual Studio oder in einem externen Browser öffnen wollen.
UMGEBUNG/SCHRIFTARTEN UND FARBEN	Legen Sie die verwendeten Schriftarten und Farbgebungen für die verschiedenen Programmelemente fest.
UMGEBUNG/START	Um die Aktualisierung der Informationen der Startseite über das Internet zu deaktivieren, entfernen Sie die Markierung INHALT HERUNTERLADEN ALLE.
TEXT-EDITOR/ALLE SPRACHEN	Zur besseren Übersicht in langen Quelltexten können Sie die Anzeige der Zeilennummern aktivieren.
TEXT-EDITOR/ALLE SPRACHEN/TABSTOPPS	Um die Einrückung im Text-Editor festzulegen, ändern Sie die Werte in den Feldern TABULATORGRÖSSE und EINZUGSGRÖSSE. Außerdem können Sie einstellen, ob für den Einzug Leerzeichen oder Tabulatoren verwendet werden sollen. Die Verwendung von Leerzeichen und einem Einzug von 2 wird in diesem Buch genutzt, da der Quelltext dadurch nicht zu »breit« und in jedem Editor gleich formatiert wird.
TEXT-EDITOR/C# / FORMATIERUNG	Bei der Eingabe von Code formatiert der Text-Editor diesen automatisch entsprechend den Angaben dieser Kategorien. Sie können z. B. einstellen, ob die öffnende geschweifte Klammer hinter einer Anweisung oder auf einer neuen Zeile eingefügt wird. Wenn Sie die einzelnen Optionen anklicken, wird im unteren Bereich eine Vorschau der Auswirkung angezeigt.

3.3 Verwendung des Text-Editors

Der Text-Editor im Visual Studio ist ein sehr mächtiges Werkzeug mit zahlreichen Features. Er hilft Ihnen in verschiedenen Situationen bei der Eingabe des Programmcodes, liefert zahlreiche visuelle Hinweise und automatisiert viele Operationen. Im Folgenden sollen einige der Merkmale kurz vorgestellt werden.

IntelliSense

Die mitdenkende Hilfe im Editor nennt sich IntelliSense. Wenn Sie eine Anweisung eingeben, wird sofort eine Liste mit den möglichen Entsprechungen angezeigt. Sie können ganz normal weiter schreiben, oder wählen einen Eintrag aus dieser Liste mit der Maus oder den Pfeiltasten aus. Über die Tastenkombination Strg Leer können Sie die Liste jederzeit wieder öffnen, sollte sie einmal nicht sichtbar sein. Setzen Sie nach der Eingabe eines vollständigen Bezeichners einen Punkt, z. B. `Console.`, werden unter anderem alle verfügbaren Methoden angezeigt, so auch die Methode `WriteLine()`. Wählen Sie eine Methode aus, wird in einem zusätzlichen Fenster eine kurze Hilfe angezeigt (Abbildung 3.8).

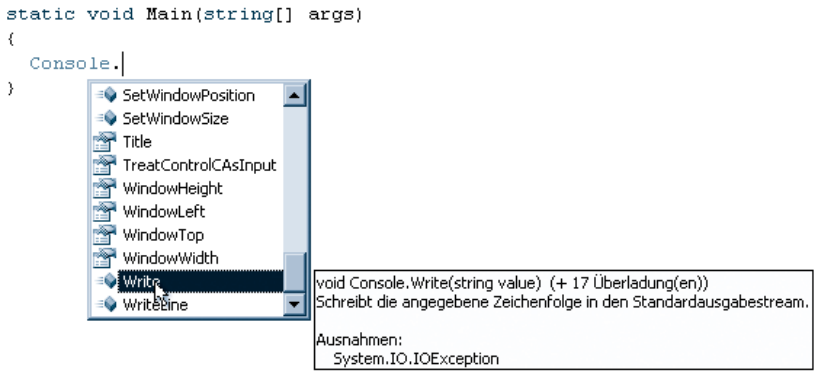


Abbildung 3.8
Vervollständigungsliste und Hilfe anzeigen

Geben Sie nach dem Methodennamen eine öffnende Klammer ein, wird die Parameterhilfe geöffnet. Links wird angezeigt (Abbildung 3.9), wie viele Überladungen (Varianten) es von der Methode gibt. Um eine andere Überladung auszuwählen, verwenden Sie die Pfeiltasten ↑ und ↓.

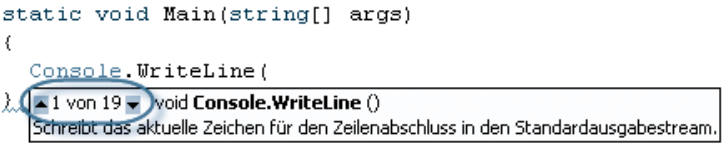


Abbildung 3.9
Parameterhilfe mit Anzeige der Überladungen

Lesezeichen

In größeren Quelltexten können Sie wichtige Stellen über Lesezeichen markieren. Verwenden Sie dazu die Untermenüpunkte von BEARBEITEN/LESEZEICHEN oder die verfügbaren Tastenkombinationen.

Hilfe anzeigen

Bewegen Sie den Mauszeiger auf ein Programmelement und warten Sie einen Moment. Liegt ein Hilfetext zu diesem Element vor, wird dieser in einem Tooltipp-Fenster als so genannte QuickInfo angezeigt.

3.4 Windows-Anwendungen

3.4.1 Eine Windows-Anwendung erstellen

Das Erstellen von grafischen Anwendungen wird ab Kapitel 22 beschrieben. Das Visual Studio bietet mit dem Windows Forms-Designer, der Toolbox und dem Eigenschaften-Fenster verschiedene Hilfsmittel, um Ihnen die Erstellung und Konfiguration einer grafischen Oberfläche zu erleichtern. An dieser Stelle soll kurz das Erstellen einer einfachen Windows-Anwendung mithilfe des Visual Studios gezeigt werden.

Hinweis

Mit dem .NET Framework 3.0 kam die Erstellung grafischer Anwendung mit der WPF (Windows Presentation Foundation) hinzu. Die Vorgehensweise ist dabei eine völlig andere, als unter Windows Forms (diese Variante gibt es seit der ersten Version des .NET Frameworks). WPF-Anwendungen lassen sich auch mit den Express Versionen erstellen. Allerdings wird dieser Anwendungstyp in diesem Buch nicht besprochen, da die Erläuterungen für eine kurze Einführung zu umfangreich sind. Ich verweise an dieser Stelle auf die entsprechende Literatur, z. B. mein eigenes Buch zur Programmierung mit der WPF.

1. Falls Sie noch ein Projekt geöffnet haben, können Sie dieses über den Menüpunkt DATEI/PROJEKTMAPPE SCHLIESSEN schließen.
2. Wenn Sie das Projekt einer bereits vorhandenen Projektmappe hinzufügen möchten, rufen Sie im Kontextmenü der Projektmappe im Projektmappen-Explorer den Menüpunkt HINZUFÜGEN/NEUES PROJEKT auf (um ein Projekt in einer neuen Projektmappe zu erstellen rufen Sie den Menüpunkt DATEI/NEUES PROJEKT auf).
3. Klicken Sie im Dialogfenster NEUES PROJEKT auf das Symbol WINDOWS FORMS-ANWENDUNG. Geben Sie im Eingabefeld NAME einen Namen für das Projekt und im Feld SPEICHERORT das Ziel der Projektdateien an. Bestätigen Sie mit OK.

In Abbildung 3.10 werden die Fenster gezeigt, die beim Erstellen einer Windows-Anwendung standardmäßig geöffnet werden. Im Projektmappen-Explorer (1) kommen neben dem Hauptprogramm *Program.cs* noch die Formulardateien *Forms.cs* und *Forms.Designer.cs* hinzu.

Im Windows Forms-Designer (2) werden die Steuerelemente angeordnet, welche die Programmoberfläche bilden sollen.

4. Die Steuerelemente befinden sich in mehreren Registerkarten in der Toolbox (3). Damit diese dauerhaft geöffnet bleibt, klicken Sie in der Titelleiste der Toolbox auf das Reiszweckensymbol. Klappen Sie eine Registerkarte auf, werden die enthaltenen Steuerelemente (auch Kontrollelemente, Komponenten) angezeigt. Diese können z. B. per Drag&Drop oder Doppelklick

in das Formular eingefügt werden. Danach können Sie die Eigenschaften anpassen, z. B. die Farbe oder die Höhe und Breite (5). Bei (6) schalten Sie zwischen der Anzeige der Eigenschaften und Ereignisse um.

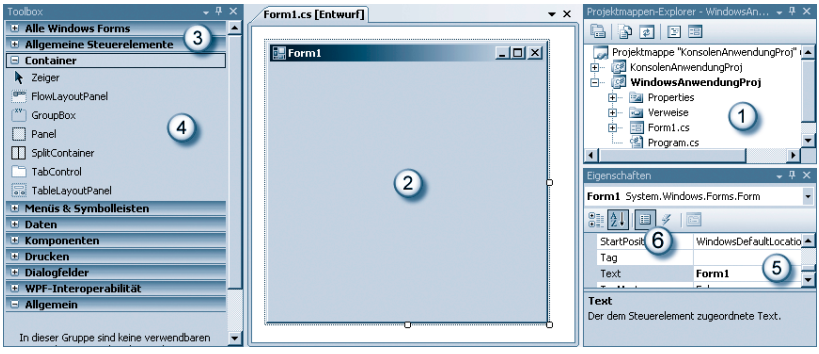


Abbildung 3.10
Fenster zum Entwickeln von Windows-Anwendungen

Die Toolbox verwenden

5. Öffnen Sie in der Toolbox das Register ALLGEMEINE STEUERELEMENTE. Fügen Sie einen `Button` und eine `TextBox` in das Formular ein. Wenn Sie die Kontrollelemente im Formular anklicken, können Sie diese verschieben oder deren Größe ändern.

Eigenschaften einstellen

6. Markieren Sie den Button.
7. Wechseln Sie in das Fenster EIGENSCHAFTEN und wählen Sie die Eigenschaft `Text` aus (linke Spalte). Geben Sie den Text `Start` ein.
8. Markieren Sie das Formular, indem Sie auf die Titelleiste klicken.
9. Ändern Sie den Wert der Eigenschaft `Text` in `Klick mich`. Nach dem Betätigen von Enter sehen Sie sofort die Änderung, in dem der neue Text in der Titelleiste des Formulars angezeigt wird.

Auf Ereignisse reagieren

10. Markieren Sie den Button.
11. Klicken Sie auf die Blitzschaltfläche im Eigenschaften-Fenster.
12. Wählen Sie das Ereignis `Click` aus und klicken Sie doppelt in die rechte Spalte neben dem Ereignisnamen. Alternativ können Sie auch doppelt auf den Button klicken.
13. Es wurde der Rumpf einer Ereignisbehandlungsroutine im Text-Editor erzeugt. Geben Sie die blau dargestellte Anweisung an der Stelle am Cursor ein:

```
private void button1_Click(object sender, EventArgs e)
{
  textBox1.Text = "Hallo";
}
```

Listing 3.2
Forms1.cs (WindowsAnwendungProj)

Anwendung ausführen

14. Drücken Sie die Tastenkombination Strg F5, um die Anwendung zu aktualisieren (zu übersetzen) und danach sofort auszuführen.
15. Klicken Sie auf den Button. Der Text `Hallo` sollte nun im Eingabefeld angezeigt werden.
16. Beenden Sie die Anwendung.

3.5 Hilfe verwenden

Kein im Handel erhältliches Buch schafft es, alle Entwicklungsmöglichkeiten, Klassen, Interfaces und Methoden zu erläutern, die Sie im .NET Framework nutzen können. Der Zugriff auf das Hilfesystem wird also unvermeidlich bleiben. Unter dem Menüpunkt HILFE verstecken sich verschiedene Zugriffsmöglichkeiten auf die Hilfe.

Tabelle 3.2 Verschiedene Anwendungsmöglichkeiten der Hilfe

Menüpunkt	Beschreibung
GEWUSST WIE	Hier gibt es Hilfestellungen zu konkreten Problemstellungen wie in einem Kochbuch (wie macht man dies und wie macht man das...).
SUCHEN	Aktiviert die Suchfunktion, welche nach einer Stichwortsuche die Fundstellen als Liste von Links anzeigt und bei einem Klick auf einen Link die Seite im Hilfefenster öffnet.
INHALT	Zeigt das Inhaltsverzeichnis aller Hilfethemen an. Dieses unterteilt sich z. B. in Themen zur Verwendung des .NET Frameworks oder der Visual C# Express Edition.
INDEX	Im Index können Sie nach Stichwörtern suchen. Erhalten Sie zu viele Fundstellen, können Sie z. B. nach einer Klasse durch das Anfügen der Zeichenfolge `-Klasse` suchen. Die Suche nach der Klasse `String` kann z. B. so erfolgen: `String-Klasse` Werden mehrere Fundstellen ermittelt, können Sie zwischen diesen in der Liste wählen, die im unteren rechten Fenster angezeigt wird.
HILFEFAVORITEN	Ein Hilfethema kann in den Bereich HILFEFAVORITEN aufgenommen werden. Sie müssen dazu das Thema rechts auswählen und auf das entsprechende Symbol in der Toolpalette klicken.

Hilfeinformationen auswerten

Haben Sie erfolgreich Hilfethemen zu einer Klasse oder einem anderen Stichwort gefunden, müssen Sie noch die richtige Hilfeseite herausfinden. Im Falle der Klasse `String` sind folgende Stichworteinträge relevant:

Stichwort	Beschreibung
String-Klasse	Beschreibt die Syntax und die Verwendung der Klasse *String*. Dies sollte der erste Anlaufpunkt sein, um eine unbekannte Klasse kennen zu lernen.
alle Member	Beschreibt alle Methoden, Eigenschaften etc. der Klasse. Über (1) und (2) in können Sie die Bereiche zu- oder aufklappen. Über (3) wählen Sie aus, ob Sie nur die öffentlichen oder auch die geschützten Member anzeigen wollen.
Informationen zu String-Klasse	Entspricht *String-Klasse*.

Tabelle 3.3
Stichworteinträge und Hilfethemen der Klasse String

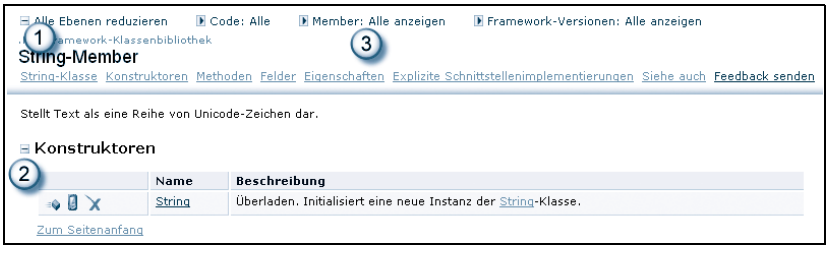


Abbildung 3.11
Hilfe zu den Membern einer Klasse

3.6 Übungsaufgaben

Aufgabe 1

Um optimal mit dem Visual Studio zu arbeiten, sollten Sie gleich zu Beginn einige Einstellungen vornehmen. Nach einiger Zeit werden sich eventuell weitere dazugesellen, wenn Sie etwas mehr Erfahrung gesammelt haben. Ändern Sie in dieser Übung den Einzug im Text-Editor auf 3 Zeichen. Setzen Sie außerdem die Schriftgröße auf 12.

Aufgabe 2

Erstellen Sie eine Konsolenanwendung. Geben Sie eine Aufforderung zur Eingabe Ihres Namens aus. Lesen Sie den Wert ein und geben Sie ihn wieder aus. Führen Sie die Anwendung einmal mit und einmal ohne Debugger aus. Was stellen Sie für Unterschiede bei der Ausführung fest?

Aufgabe 3

Ändern Sie in der Windows-Beispielanwendung in diesem Kapitel die Hintergrundfarbe des Formulars auf rot sowie die Breite und Höhe auf 400 und 250. Versuchen Sie die dazu notwendigen Eigenschaften im Eigenschaften-Fenster selbst zu ermitteln. Dies wird anfangs etwas länger dauern, aber nur auf diese Weise entdecken Sie, welche Möglichkeiten im .NET Framework stecken.