Das erste Programm – die Benutzeroberfläche

Bisher ist unser erstes Programm nicht gerade geeignet, Freunde und Bekannte zu beeindrucken. Außer einem leeren Fenster, das man minimieren und schließen kann, hat es kaum etwas zu bieten. Dies wollen wir nun ändern und uns dabei gleichzeitig mit den typischen Arbeitsabläufen bei der komponentengestützten Programmierung vertraut machen.

Die Designer-Umgebung 3.1

Zur Erstellung und Bearbeitung der grafischen Benutzerschnittstelle (sprich der Fenster) stellt Visual Studio im Wesentlichen vier Elemente zur Verfügung:

- den Windows Forms-Designer,
- den Werkzeugkasten,
- das Eigenschaftenfenster
- und das Menü *Format*

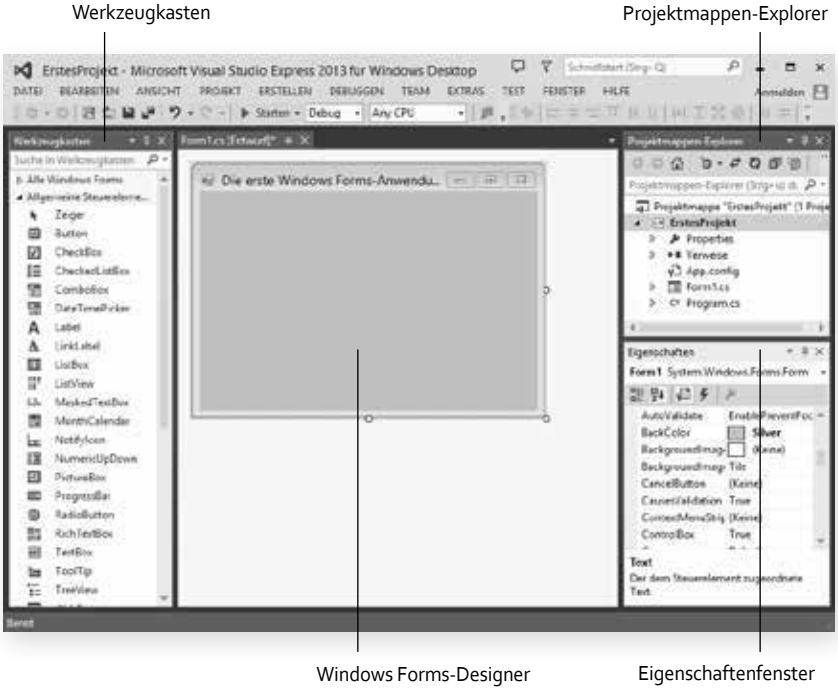

*Abbildung 3.1: **Bearbeitung eines Fensters in Visual Studio***

GUI

Aus Sicht eines Anwenders bestehen Windows-Anwendungen nicht aus Code oder Programmdateien, sondern aus den Fenstern der Anwendung. Die Gesamt dieser Fenster bezeichnen wir als die *grafische Benutzerschnittstelle*, kurz GUI.

Der Windows Forms-Designer

1. Falls Sie das Projekt aus *Kapitel 2* zwischenzeitlich geschlossen haben, laden Sie es jetzt noch einmal in Visual Studio. Rufen Sie dazu den Befehl *Datei/Projekt öffnen* auf und wählen Sie die Projektmappendatei aus dem Projektverzeichnis des ausführenden Programms (z.B. *Beispiele\Kapo2\ErstesProjekt.sln* oder *Beispiele\Kapo2\ErstesProjekt_nachStudie_1/ErstesProjekt.sln*, wenn Sie das Projekt aus der Beispielsammlung verwenden möchten) aus.
2. Doppelklicken Sie rechts im Projektmappen-Explorer auf den Knoten des Hauptfensters (Form1.cs), um das Fenster zur Bearbeitung in den Designer zu laden.
3. Verändern Sie die Abmaße des Fensters so, dass es etwas breiter und schmaler wird.

   Am rechten und unteren Rand des Fensters sehen Sie weiße Markierungskästchen. Ziehen Sie diese mit der Maus, um die Maße des Fensters zu verändern.

Der Werkzeugkasten

Im Werkzeugkasten finden Sie eine Auswahl an nützlichen Komponenten (fertigen Software-Bausteinen), die Sie direkt mit der Maus aufnehmen und in Ihr Anwendungsfenster im Windows Forms-Designer einfügen können. Ein Großteil der Komponenten, die bereits in den .NET-Bibliotheken vordefiniert sind, repräsentieren Steuerelemente (Schaltflächen, Listenfelder, Optionsfelder etc.). Komponenten können aber grundsätzlich beliebigen Aufgaben gewidmet sein und sie müssen nicht einmal über eine sichtbare Oberfläche verfügen.

Der Werkzeugkasten befindet sich anfangs minimiert am linken Rand.

4. Fahren Sie einmal mit der Maus über das Symbol; Sie werden feststellen, dass der Werkzeugkasten ganz von allein aufspringt. Allerdings wird der Werkzeugkasten auch direkt wieder geschlossen, sobald Sie den Bereich der eingeblendeten Optionen wieder verlassen. Wenn Sie lieber mit einem Werkzeugkasten arbeiten, der permanent im Vordergrund liegt, klicken Sie rechts oben in der Titelleiste des Werkzeugkastens auf das mittlere Symbol, das eine nach links zeigende Heftzwecke darstellt. Daraufhin weist die Heftzwecke nach unten und der Werkzeugkasten steht jederzeit für den Zugriff bereit. Allerdings verkleinert sich dadurch das Designer-Fenster. Sie können den Werkzeugkasten durch einen Klick auf die Heftzwecke jederzeit wieder minimieren.

Der Werkzeugkasten kann auch jederzeit über den Menübefehl *Ansicht/Werkzeugkasten* geöffnet werden.

Werkzeugkasten

Das Eigenschaftenfenster

Das Eigenschaftenfenster wird geöffnet, wenn Sie im Kontextmenü des Designers den Befehl *Eigenschaften* auswählen. Alternativ können Sie auch im Menü *Ansicht* den Befehl *Eigenschaftenfenster* aufrufen.

5 Lassen Sie das Eigenschaftenfenster anzeigen. Falls nötig, platzieren Sie das Eigenschaftenfenster auf der rechten Seite unterhalb des Projektmappen-Explorers.

Visual Studio hilft Ihnen beim Positionieren und Andocken des Fensters durch Einblendung von kleinen Positionierungsfenstern. Die Positionierungsfenster erscheinen, sobald Sie das Fenster mit der Maus ziehen. Wenn Sie das Fenster über eines der Positionierungsfenster bewegen, markiert Visual Studio den Bereich, in den das Fenster angedockt würde. Haben Sie die korrekte Andockposition gefunden, müssen Sie das Fenster nur noch ablegen (Maustaste loslassen)

Das Andock-System von Visual Studio

Über das Eigenschaftenfenster können Sie die Komponenten, die im Designer angezeigt werden (einschließlich des übergeordneten Formulars), konfigurieren.

Zu diesem Zweck ist das Eigenschaftenfenster als zweispaltige Liste aufgebaut. In der linken Spalte werden die Felder der ausgewählten Komponente aufgelistet, in der rechten Spalte stehen passende Eingabefelder, über die den Feldern Werte zugewiesen werden können.

Mit den ersten beiden Schaltflächen aus der Symbolleiste des Eigenschaftenfensters können Sie die Liste der Eigenschaften nach Kategorien oder alphabetisch geordnet anzeigen lassen.

Sortierung

Hinter den Komponenten verbergen sich letztlich Klassendefinitionen. Das Besondere an diesen Klassen ist, dass sie einen Teil ihrer Felder (Datenelemente) als öffentliche *Eigenschaften* oder *Ereignisse* definieren. Wie dies syntaktisch aussieht, muss uns jetzt noch nicht interessieren. Wichtig ist, dass diese Eigenschaften und Ereignisse im Eigenschaftenfenster angezeigt und bearbeitet werden können. (Per Voreinstellung sehen Sie im Eigenschaftenfenster die Eigenschaften der Komponente. Durch Klick auf das Blitz-Symbol in der Symbolleiste des Eigenschaftenfensters können Sie zu den Ereignissen wechseln.)

Komponenten

6. Markieren Sie das Fenster, indem Sie es im Windows Forms-Designer anklicken.

7. Klicken Sie im Eigenschaftenfenster in das Eingabefeld neben der Eigenschaft *Text*. Überschreiben Sie dort den automatisch erzeugten Wert *Form1* mit **Die erste Windows Forms-Anwendung**.

Sobald Sie die Eingabetaste drücken oder zu einer anderen Eigenschaft wechseln, wird der von Ihnen eingegebene Wert für die Titelleiste Ihres Anwendungsfensters im Designer sichtbar.

8. Ändern Sie die Hintergrundfarbe. Klicken Sie in das Feld neben der Eigenschaft *BackColor* und geben Sie als neue Farbe **Silver** ein. (Oder klicken Sie auf die kleine Schaltfläche mit dem abwärts gerichteten Pfeil und wählen Sie in dem aufspringenden Dialog auf der Seite *Web* den Eintrag für *Silver* aus.)

Das Menü *Format*

Wenn Sie im Windows Forms-Designer arbeiten, wird die Menüleiste von Visual Studio um ein zusätzliches Menü *Format* erweitert, das spezielle Befehle zum Arrangieren der Komponenten im Fenster enthält (mehr dazu in Kürze). Alternativ können die Befehle im Menü *Format* auch über die Symbolschaltflächen in der Symbolleiste *Layout* aufgerufen werden.

3.2 Komponenten aufnehmen

Um Ihr Anwendungsfenster mit Komponenten – vorzugsweise Steuerelementen – zu bestücken, müssen Sie die gewünschten Komponenten lediglich aus dem Werkzeugkasten, der sich am linken Rand befindet, in das Fenster zu ziehen.

9. Falls nicht bereits geschehen, pinnen Sie jetzt den Werkzeugkasten mithilfe des Heftzwecken-Symbols fest, damit er geöffnet bleibt. Achten Sie darauf, dass der Bereich *Allgemeine Steuerelemente* angezeigt wird.

10. Klicken Sie auf die Beschriftung-Komponente *Label* und halten Sie die Maustaste gedrückt.

    Übrigens: Wenn Sie den Mauszeiger etwas länger über einer Komponente stehen lassen, wird als *QuickInfo* ein kleiner Hilfetext und der Name der Komponente angezeigt.

11. Ziehen Sie die Komponente per Drag&Drop in Ihr Anwendungsfenster.

    Visual Studio fügt daraufhin eine Instanz der Komponente in das Fenster ein.

Einfügen per Drag&Drop

Wenn Sie Komponenten per Drag&Drop einfügen, werden ab der zweiten Komponente Ausrichtungslinien im Anwendungsfenster eingeblendet, die Ihnen helfen, die einzelnen Komponenten optimal zueinander zu platzieren.

Abbildung 3.2: Das Anwendungsfenster mit einem Beschriftungsfeld

⑫ Testen Sie das Programm (Strg F5).

Bevor wir das eingefügte Beschriftungsfeld weiter anpassen, wollen wir kurz analysieren, was eigentlich geschehen ist.

Nachdem wir das Beschriftungsfeld in das Fenster aufgenommen hatten, wurde es sofort im Designerfenster angezeigt. Aber auch der Quelltext der Datei *Form1.Designer.cs* wurde aktualisiert und um eine Zeile erweitert. Doppelklicken Sie im Projektmappen-Explorer auf *Form1.Designer.cs* (befindet sich unter dem Knoten *Form1.cs*).

Neu hinzugekommen ist hier die letzte Zeile:

```
private System.Windows.Forms.Label label1;
```

Als Element der Klasse `Label` (aus dem Namespace `System.Windows.Form`) wird hier eine Variable namens `label1` definiert.

Wie Sie wissen, stehen hinter den Komponenten aus dem Werkzeugkasten letztlich Klassen (für Beschriftungsfelder heißt diese Klasse, wenig überraschend, `Label`). Wenn wir also ein Beschriftungsfeld aus dem Werkzeugkasten auswählen und per Mausklick in ein Fenster einfügen, richtet Visual Studio eine Variable vom Typ der Klasse `Label` ein.

Dies ist allerdings nicht die einzige Änderung.

Expandieren Sie im Editor den Knoten *Von Windows Form-Designer generierter Code*.

Jetzt sehen Sie den Inhalt der Methode `InitializeComponent()`. Hier fügt der Designer sämtlichen Code zur Erzeugung und Konfiguration des Fensters und seiner Komponenten ein. Ganz zuoberst sehen Sie den Aufruf des `Label`-Konstruktors, der unser Beschrif-

tungsfeld erzeugt. Darunter folgen zwei Abschnitte, in denen das Beschriftungsfeld und das Fenster konfiguriert werden. Sehen Sie sich die Anweisungen für das Fenster etwas genauer an und Sie können sehen, wie die Einstellungen zu Fenstergröße, Titel und Hintergrundfarbe vom Designer in Quelltext verwandelt wurden.

```
namespace ErstesProjekt
{
    partial class Form1
    {
        ...
        #region Vom Windows Form-Designer generierter Code

        /// <summary>
        /// Erforderliche Methode für die Designerunterstützung.
        /// Der Inhalt der Methode darf nicht mit dem Code-Editor
        /// geändert werden.
        /// </summary>
        private void InitializeComponent()
        {
            this.label1 = new System.Windows.Forms.Label();
            this.SuspendLayout();
            //
            // label1
            //
            this.label1.AutoSize = true;
            this.label1.Location = new System.Drawing.Point(64, 59);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(35, 13);
            this.label1.TabIndex = 0;
            this.label1.Text = "label1";
            //
            // Form1
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F,
                                                                13F);
            this.AutoScaleMode =
                            System.Windows.Forms.AutoScaleMode.Font;
            this.BackColor = System.Drawing.Color.Silver;
            this.ClientSize = new System.Drawing.Size(379, 246);
            this.Controls.Add(this.label1);
            this.Name = "Form1";
            this.Text = "Die erste Windows Forms-Anwendung";
            this.ResumeLayout(false);
```

```
            this.PerformLayout();
        }

        #endregion

        private System.Windows.Forms.Label label1;
    }
}
```

Listing 3.1: Auszug aus Form1.Designer.cs

Finger weg von der Designer-Datei

Denken Sie daran, dass der Code aus *Form1.Designer.cs* nur vom Designer bearbeitet werden sollte. Wenn Sie einen anderen Titel oder eine andere Hintergrundfarbe wählen möchten oder sonstige Änderungen an dem Layout des Fensters und seiner Komponenten vornehmen möchten, benutzen Sie dazu den Designer und das Eigenschaftenfenster.

3.3 Komponenten konfigurieren

Wechseln Sie jetzt noch einmal zurück in die Entwurfsansicht unseres Anwendungsfensters.

⑬ Markieren Sie das Beschriftungsfeld. Klicken Sie dazu das Beschriftungsfeld im Designer-Fenster an.

Im Eigenschaftenfenster auf der rechten Seite unten werden daraufhin die Eigenschaften des Textfeldes angezeigt.

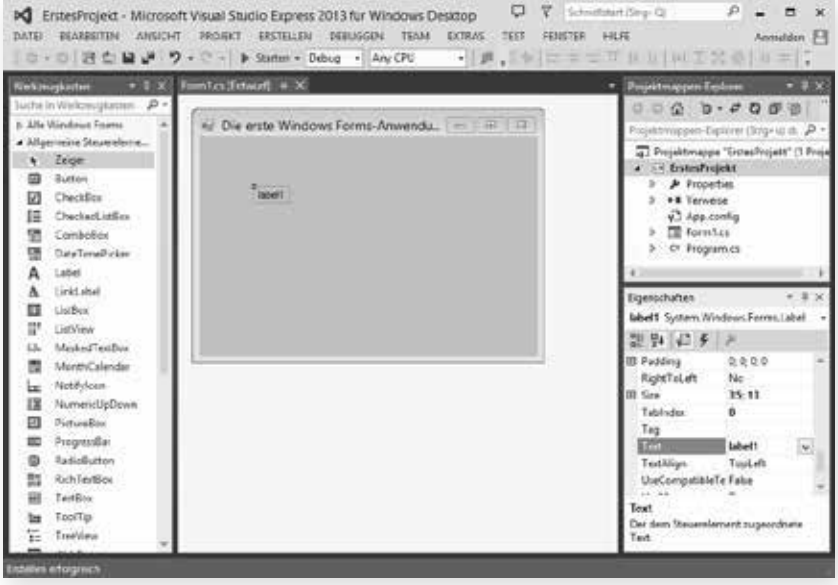

Abbildung 3.3: Bearbeiten der Eigenschaften einer Komponente

14. Ändern Sie den Text des Beschriftungsfeldes. Klicken Sie dazu in das Eingabefeld rechts neben der Eigenschaft *Text* und geben Sie als neuen Text **Hallo Welt!** ein.

15. Ändern Sie den Font des Beschriftungsfeldes. Klicken Sie dazu links neben der Eigenschaft *Font* auf das Pfeilsymbol, das Ihnen verrät, dass sich hinter *Font* noch Unterpunkte verbergen. Setzen Sie die Größe der Schriftart in dem Eingabefeld rechts der Unter-Eigenschaft *Size* auf **24**.

Da die *AutoSize*-Eigenschaft auf *True* gesetzt ist, wird das Beschriftungsfeld bei einer Vergrößerung der Schriftart automatisch von der Größe her angepasst. Wenn Sie das Beschriftungsfeld unabhängig von der Schrift vergrößern möchten, müssen Sie erst diese Eigenschaft auf *False* setzen.

16. Scrollen Sie dazu im Eigenschaftenfenster zur Eigenschaft *AutoSize*. Klicken Sie im Eingabefeld auf den abwärts gerichteten Pfeil und wählen Sie in der Liste **False** aus.

Als Folge werden am Rahmen Ihres Beschriftungsfeldes acht Markierungspunkte sichtbar, mit deren Hilfe Sie die Größe durch Ziehen verändern können.

17. Klicken Sie mit der linken Maustaste auf einen der Punkte des Markierungsrahmens des Beschriftungsfeldes und ziehen Sie den Rahmen auf.

Schön wäre es auch, wenn der Schriftzug horizontal zentriert im Fenster zu lesen wäre. Dazu müssen Sie zuerst das Beschriftungsfeld im Fenster und dann den Schriftzug im Beschriftungsfeld horizontal zentrieren.

18. Positionieren Sie das Beschriftungsfeld zuerst in der Vertikalen. Da es hierbei nicht auf eine exakte mittige Positionierung ankommt, ziehen Sie die Komponente einfach mit der Maus. Wählen Sie eine Position in der oberen Fensterhälfte.

19. Zum Zentrieren des Beschriftungsfeldes markieren Sie es im Designer und wählen dann im Menü *Format* den Befehl *Auf Formular zentrieren/Horizontal* aus. (Alternativ können Sie in der Layout-Symbolleiste auf das Symbol *Horizontal zentrieren* klicken.)

20. Zum Zentrieren des Schriftzugs im Beschriftungsfeld scrollen Sie im Eigenschaftenfenster zu der Eigenschaft *TextAlign* und wählen im Eingabefeld das Symbol für die mittige Ausrichtung am oberen Rand aus.

Vererbte Eigenschaften

Das Beschriftungsfeld übernimmt als Hintergrundfarbe automatisch die Farbe des Anwendungsfensters, sodass sich der Text unauffällig in das Anwendungsfenster einfügt. Wenn Sie dem Steuerelement lieber eine eigene Hintergrundfarbe zuweisen möchten, tun Sie dies wie im Falle des Fensters über die Eigenschaft *BackColor*.

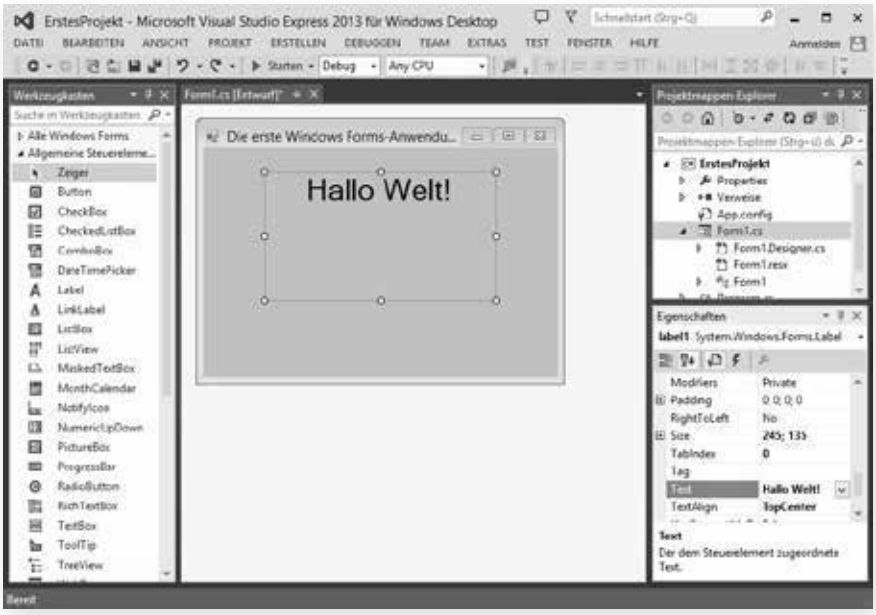

Abbildung 3.4: Das Anwendungsfenster mit konfiguriertem Beschriftungsfeld

3.4 Ereignisse bearbeiten

Komponenten verfügen nicht nur über Eigenschaften, sondern auch über Ereignisse. Die Ereignisse des Label-Steuerelements sind allerdings ziemlich uninteressant für uns. Wir erweitern unsere Anwendung daher um eine Schaltfläche und fangen das Drücken der Schaltfläche ab.

21 Fügen Sie unter dem Beschriftungsfeld eine Schaltfläche (*Button*) ein.

22 Ändern Sie den Titel der Schaltfläche (Eigenschaft *Text*) in **Klick mich** und zentrieren Sie die Schaltfläche horizontal im Fenster.

Jetzt wollen wir festlegen, was geschehen soll, wenn der Anwender die Schaltfläche drückt. Dazu müssen wir im Eigenschaftenfenster von den Eigenschaften zu den Ereignissen wechseln.

23 Klicken Sie in der Symbolleiste des Eigenschaftenfensters auf das Symbol für Ereignisse, das einen Blitz darstellt. Es erscheint eine Liste von verschiedenen Ereignissen, die man bearbeiten kann. (Achten Sie darauf, dass die Schaltfläche markiert ist, damit Sie auch tatsächlich die Ereignisse der Schaltfläche angezeigt bekommen.) Von diesen Ereignissen interessiert uns im Moment nur *Click*, da dieses Ereignis ausgelöst wird, wenn der Anwender auf die Schaltfläche klickt.

Hilfe zu Ereignissen

Wenn Sie sich näher über ein Ereignis informieren möchten, markieren Sie im Eigenschaftenfenster das Ereignis und drücken Sie F1.

24 Doppelklicken Sie in das Eingabefeld neben dem *Click*-Ereignis.

Der Windows Forms-Designer legt daraufhin eine neue Methodendefinition an und wechselt in die Code-Ansicht der Datei *Form1.cs*, wo Sie die Methode direkt bearbeiten können.

```
private void button1_Click(object sender, EventArgs e)
{

}
```

Gleichzeitig setzt er Code auf, der dafür sorgt, dass beim Anklicken der Schaltfläche automatisch die betreffende Methode aufgerufen und ausgeführt wird. Dieser Code soll uns aber im Moment nicht weiter interessieren, da für uns keine Notwendigkeit besteht, diesen Code zu verändern. Was wir aber tun wollen, ist die Methode zu implementieren, die beim Drücken der Schaltfläche aufgerufen wird.

3.5 Eigenschaften zur Laufzeit ändern

Visual Studio hat uns bereits zur Definition der Ereignisbehandlungsmethode geführt.

```
private void button1_Click(object sender, EventArgs e)
{

}
```

Wir müssen jetzt nur noch zwischen die geschweiften Klammern die Anweisungen einfügen, die abgearbeitet werden sollen, wenn das Ereignis eintritt. Doch was sollen wir hineinschreiben? Wir haben uns doch noch gar nicht näher mit C# und der C#-Syntax beschäftigt?

Nun, das ist auch gar nicht unbedingt notwendig. Wir begnügen uns einfach mit der Veränderung von Komponenten zur Laufzeit und schauen uns die benötigten Techniken vom Designer ab.

Was macht der Windows Forms-Designer, wenn wir den Text der Label-Komponente im Eigenschaftenfenster ändern? Richtig, er weist der `Text`-Eigenschaft der Komponente den neuen Text zu. Das können wir auch.

25 Implementieren Sie die Ereignisbehandlungsmethode:

Zuerst müssen Sie auf das Objekt zugreifen, das Sie bearbeiten möchten. Dies geschieht über den Variablennamen, den Visual Studio für das Objekt definiert hat. Im Falle unseres Beschriftungsfeldes wäre dies also ... Hmm, wie heißt die Variable für das Beschriftungsfeld gleich noch einmal?

Wenn Ihnen der Name nicht einfällt, klicken Sie im Designer-Fenster auf die Komponente und sehen Sie dann im Listenfeld am oberen Rand des Eigenschaftenfensters nach. Dort wird in Fettschrift der Name der Variable angezeigt: `label1`.

An den Variablennamen hängen Sie einen Punkt an. Für den C#-Compiler heißt dies, dass Sie auf ein Feld, eine Eigenschaft oder eine Methode des Objekts zugreifen möchten.

```
private void button1_Click(object sender, EventArgs e)
{
      label1.
}
```

An diesem Punkt springt Ihnen Visual Studio zur Hilfe. Visual Studio weiß, dass es sich bei `label1` um ein Objekt der Klasse `Label` handelt, und zeigt Ihnen eine Liste der Elemente dieser Klasse an.

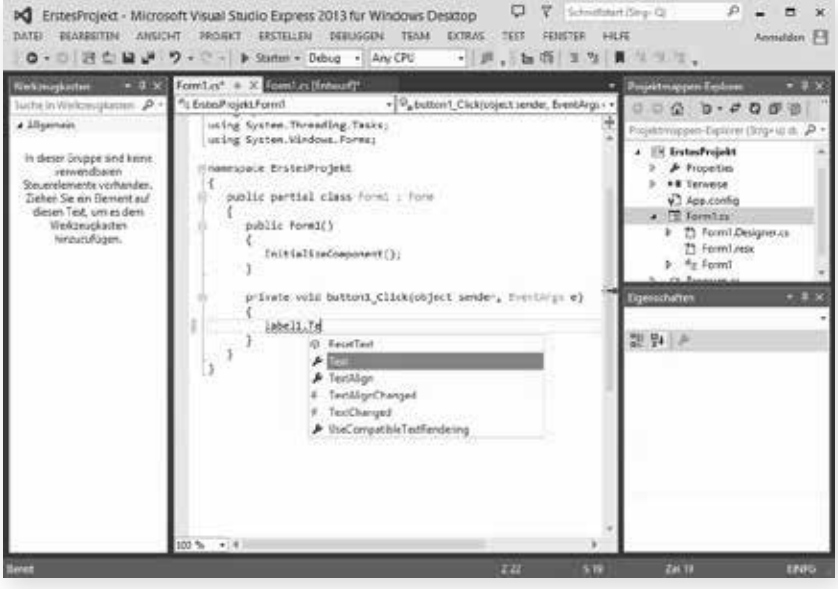

Abbildung 3.5: Automatische Memberauswahl im Visual Studio-Editor

26 Scrollen Sie in dem Popup-Fenster nach unten bis zur Eigenschaft *Text*. Alternativ können Sie auch die ersten Buchstaben der Eigenschaft eintippen. Doppelklicken Sie dann auf den Eintrag.

Anschließend müssen Sie für die Eigenschaft nur noch den neuen Text angeben. Dazu geben Sie nach einem Gleichheitszeichen den gewünschten neuen Text in Anführungszeichen ein (Semikolon am Ende nicht vergessen!):

```
private void button1_Click(object sender, EventArgs e)
{
      label1.Text = "Auf Wiedersehen!";
}
```

27 Führen Sie das Programm erneut aus (Strg F5).

Abbildung 3.6: ***Das Programm nach Drücken der Schaltfläche***

Text unvollständig?

Sollte der Text »Auf Wiedersehen!« nicht vollständig angezeigt werden, liegt dies daran, dass das Beschriftungsfeld zu klein ist. Vergrößern Sie es einfach durch Aufziehen des Markierungsrahmens in Höhe und/oder Breite im Designer-Fenster und zentrieren Sie es danach noch einmal.

3.6 Die Eigenschaft Name

Wie Sie bereits wissen, erzeugt Visual Studio für jede Komponente, die Sie aus dem Werkzeugkasten in Ihre Anwendung einfügen, eine passende Variable. Der Name dieser Variablen kann über die Eigenschaft *Name* im Eigenschaftenfenster geändert werden. (Sie finden sie in der Kategorie *Entwurf* bzw. bei alphabetischer Sortierung ganz oben, da sie als *(Name)* aufgeführt wird.) Visual Studio ersetzt dann im Quelltext der aktuellen Datei alle Vorkommen des alten Namens durch den neuen Namen.

Was tut sich auf der Festplatte? 3.7

Zum Abschluss dieses Kapitels wollen wir uns noch ansehen, wie sich die von uns erzeugten Projekte auf der Festplatte präsentieren.

Erinnern Sie sich noch, unter welchem Namen und an welchem Ort Sie das Projekt angelegt haben? Suchen Sie jetzt im Windows Explorer unter dem angegebenen Pfad nach diesem Verzeichnis.

In dem Projektverzeichnis hat Visual Studio unter anderem folgende Dateien abgespeichert:

- die Projektdatei mit der Dateiendung *.csproj*, in der alle wichtigen Informationen zu Aufbau und Konfiguration des Projekts, wie z.B. die zugehörigen Quelltextdateien, die verwendeten Assemblies oder die Compiler-Optionen festgehalten sind
- die Quelltextdateien (Dateiendung *.cs*)
- sowie weitere Unterverzeichnisse, um u.a. die *.exe*-Dateien aufzunehmen

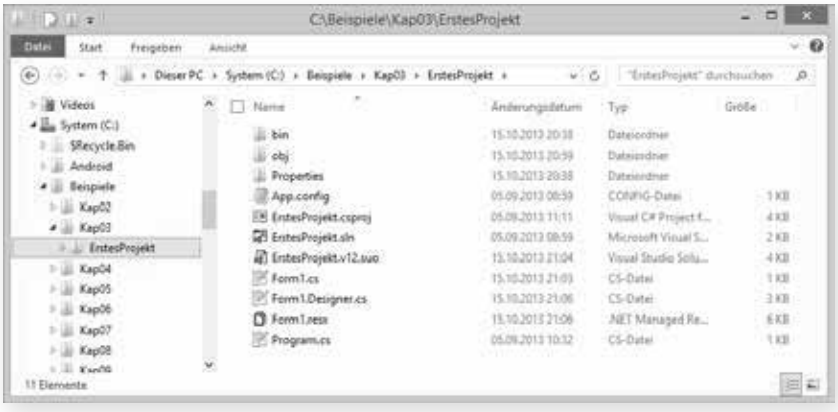

Abbildung 3.7: C#-Projekte auf der Festplatte

Anwendungen außerhalb von Visual Studio aufrufen 3.8

Natürlich wäre der Wert Ihrer C#-Anwendungen recht beschränkt, wenn man zur Ausführung der Programme stets zuerst Visual Studio starten müsste. Nun, Visual Studio ist selbstverständlich nicht erforderlich, es reicht, wenn auf dem System, auf dem die Programme ausgeführt werden, das .NET Framework vorhanden ist (dazu mehr im *Anhang C*). Dann müssen Sie einfach nur die *.exe*-Datei der Anwendung aufrufen.

Die *.exe*-Dateien Ihrer Anwendungen stehen anfangs in den von Visual Studio angelegten Projektverzeichnissen, genauer gesagt in dem jeweiligen Unterverzeichnis *bin\Debug*. Von hier aus können sie auf verschiedene Weisen aufgerufen werden.

Aufruf über den Windows Explorer

1. Öffnen Sie den Windows Explorer und navigieren Sie zu dem Projektverzeichnis der Anwendung.
2. Expandieren Sie das Unterverzeichnis *bin\Debug*.
3. Doppelklicken Sie auf die *.exe*-Datei der Anwendung.

Aufruf über Start/Ausführen bzw. Start/<Suche starten>

Programme, deren *.exe*-Datei in einem Verzeichnis aus dem Systempfad stehen, können auch über den *Ausführen*-Dialog (Aufruf mit der Tastenkombination [WinBef] [R]) gestartet werden. Welche Verzeichnisse stehen im Systempfad?

Fast immer steht dort das Windows-Unterverzeichnis *system32*. Sie müssen also nur eine Kopie Ihrer *.exe*-Datei in diesem Unterverzeichnis abzulegen, und schon können Sie die Anwendung aufrufen, indem Sie im *Ausführen*-Dialog den Namen der Anwendung (genauer gesagt der *.exe*-Datei) eingeben.

Anstatt das *system32*-Verzeichnis zu füllen, können Sie auch das *bin\Debug*-Verzeichnis der Anwendung in den Systempfad eintragen.

1. Öffnen Sie dazu unter Windows 7/Vista das Start-Menü und klicken Sie mit der rechten Maustaste auf den Eintrag *Computer*. Rufen Sie im Kontextmenü den Befehl *Eigenschaften* auf und klicken Sie im aufspringenden Dialog auf *Erweiterte Systemeinstellungen* und dann auf die Schaltfläche *Umgebungsvariablen*.

   Unter Windows 8 drücken Sie die Tastenkombination [WinBef] [W], tippen den Suchbegriff **Umgebung** ein und klicken auf den Link *Systemumgebungsvariablen bearbeiten*. Im erscheinenden Dialogfeld klicken Sie auf die Schaltfläche *Umgebungsvariablen*.
2. Wählen Sie im Bereich *Systemvariablen* den Eintrag für *Path* aus und drücken Sie die Schaltfläche *Bearbeiten*.
3. Jetzt können Sie den neuen Verzeichnispfad an das Ende des Eintrags zu *Wert der Variablen* anhängen.

   Achtung! Die einzelnen Verzeichnispfade in *Path* müssen durch Semikolon getrennt sein. Der neue Wert von *Path* könnte also beispielsweise wie folgt aussehen:

%SystemRoot%\system32;%SystemRoot%;C:\Beispiele\Kapo3\ErstesProjekt\bin\Debug

Müllen Sie den Systempfad nicht zu!

Nutzen Sie diese Option nur für ausgesuchte Anwendungen oder sammeln Sie diese Anwendungen in einem Verzeichnis, da unnötig lange *Path*-Einträge die Performance des Rechners beeinträchtigen.

Aufruf über Start- oder Desktop-Link

Die bequemste Art des Aufrufs ist sicherlich der Aufruf über das Start-Menü oder ein Desktop-Symbol.

1. Öffnen Sie den Windows Explorer und navigieren Sie zu dem Projektverzeichnis der Anwendung.
2. Ziehen Sie die *.exe*-Datei der Anwendung mit gedrückt gehaltener Maustaste auf die *Start*-Fläche. Warten Sie, bis das *Start*-Menü aufspringt, und legen Sie die *.exe*-Datei dann in der Link-Liste links oben oder im *Programme*-Menü ab.

   Oder ziehen Sie die *.exe*-Datei der Anwendung mit gedrückt gehaltener Maustaste über eine freie Fläche des Desktops. Drücken Sie zusätzlich die Tasten Strg ⇧ und lassen Sie dann die Maustaste los.

Übungen zu diesem Kapitel 3.9

Die Antworten und Lösungen zu diesen Übungen finden Sie am des Ende des Buchs.

Übung 3.1

Schreiben Sie die Ereignisbehandlungsmethode zu der Schaltfläche `button1` so um, dass beim Drücken des Schalters der Titel des Schalters in das Beschriftungsfeld kopiert wird.

Übung 3.2

1. Viele der kleineren Beispielanwendungen, denen Sie in den nachfolgenden Kapiteln begegnen werden, verfügen über eine weitgehend gleiche Benutzeroberfläche: eine Anzeigekomponente, meist ein Beschriftungsfeld (`Label`) oder ein Eingabefeld (`TextBox`), und eine Schaltfläche (`Button`), bei deren Drücken der im Buchtext besprochene Code ausgeführt und etwaige Ausgaben in die Anzeigekomponente geschrieben werden (siehe Abbildung 3.8).
2. Versuchen Sie als Vorbereitung auf die nächsten Kapitel ein Projekt mit einem solchem Hauptfenster nachzustellen. Verwenden Sie als Anzeigekomponente ein Beschriftungsfeld (`Label`) und nennen Sie es `lb_Ausgabe`. Die Schaltfläche nennen Sie `btn_Ausfuehren`. Geben Sie als Antwort auf das Drücken der Schaltfläche einfach einen Grußtext in das Beschriftungsfeld aus.

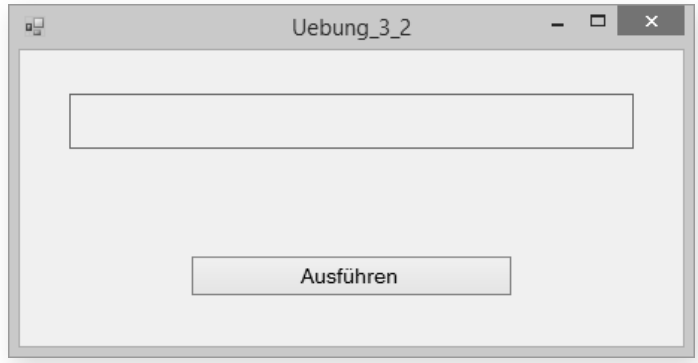

Abbildung 3.8: ***Die typische Oberfläche eines Demo-Progamms***