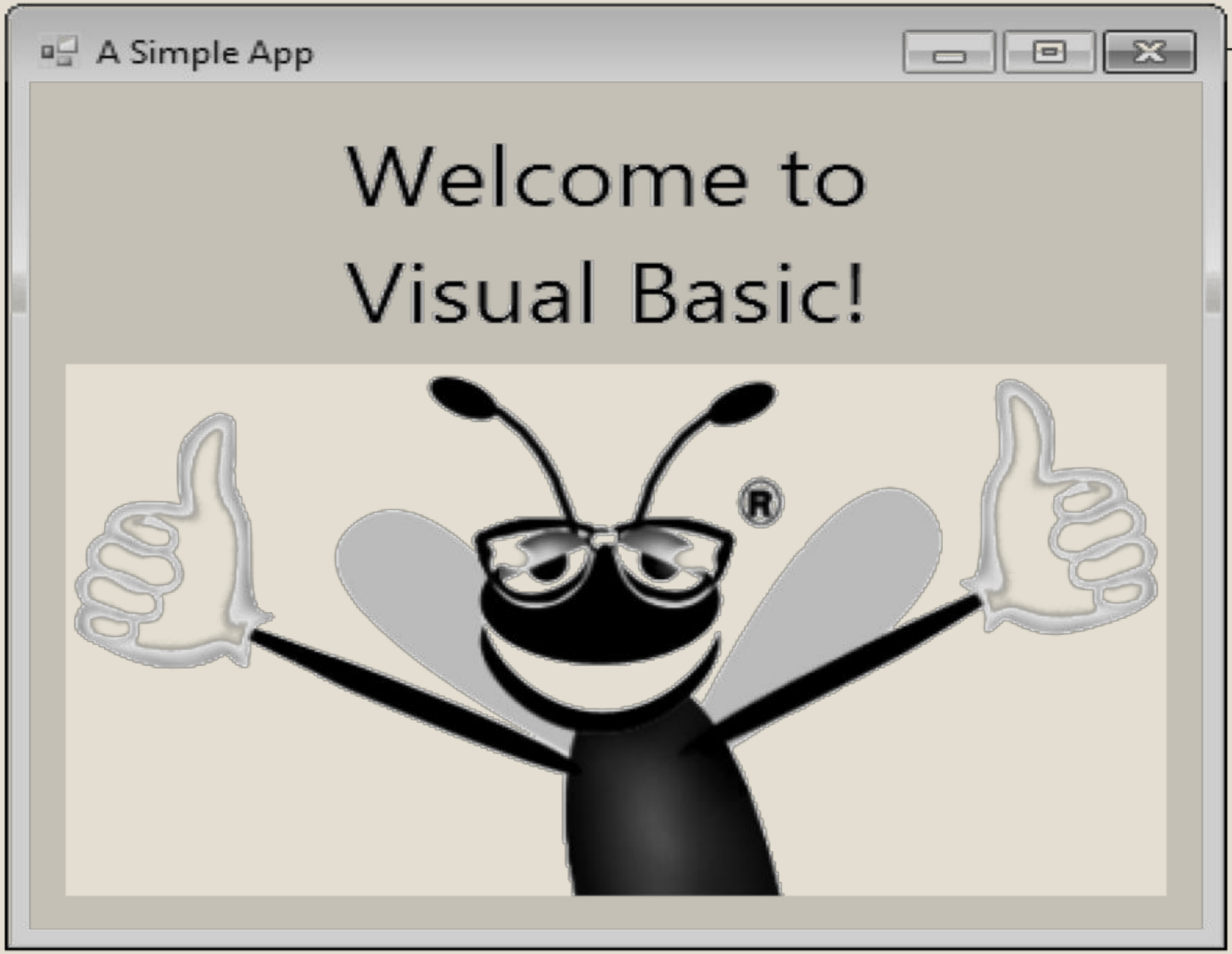
Microsoft Visual Basic 2010 Creating an Application:

Introduction

- **Visual Studio® 2010** is Microsoft's **integrated development environment (IDE)** for creating, running and debugging apps written in various .NET programming languages.
- This Lecture overviews the Visual Studio 2010 IDE and shows how to create a simple Visual Basic app by dragging and dropping predefined building blocks into place—a technique known as **visual app development**.



Welcome to
Visual Basic!

Close box

Figure 2.2 **Welcome** app executing.

2.2 Overview of the Visual Basic 2010 Express Edition IDE

- There are several versions of Visual Studio.
- The examples in this lecture are based on the *Microsoft Visual Basic 2010 Express*.
- Microsoft also offers a full version of Visual Studio 2010, which includes support for Visual Basic and other languages, such as Visual C# and Visual C++.
- To start Microsoft Visual Basic 2010 Express Edition, select **Start > All Programs > Microsoft Visual Studio 2010 Express > Microsoft Visual Basic 2010 Express**.

2.2 Overview of the Visual Basic 2010 Express Edition IDE (Cont.)

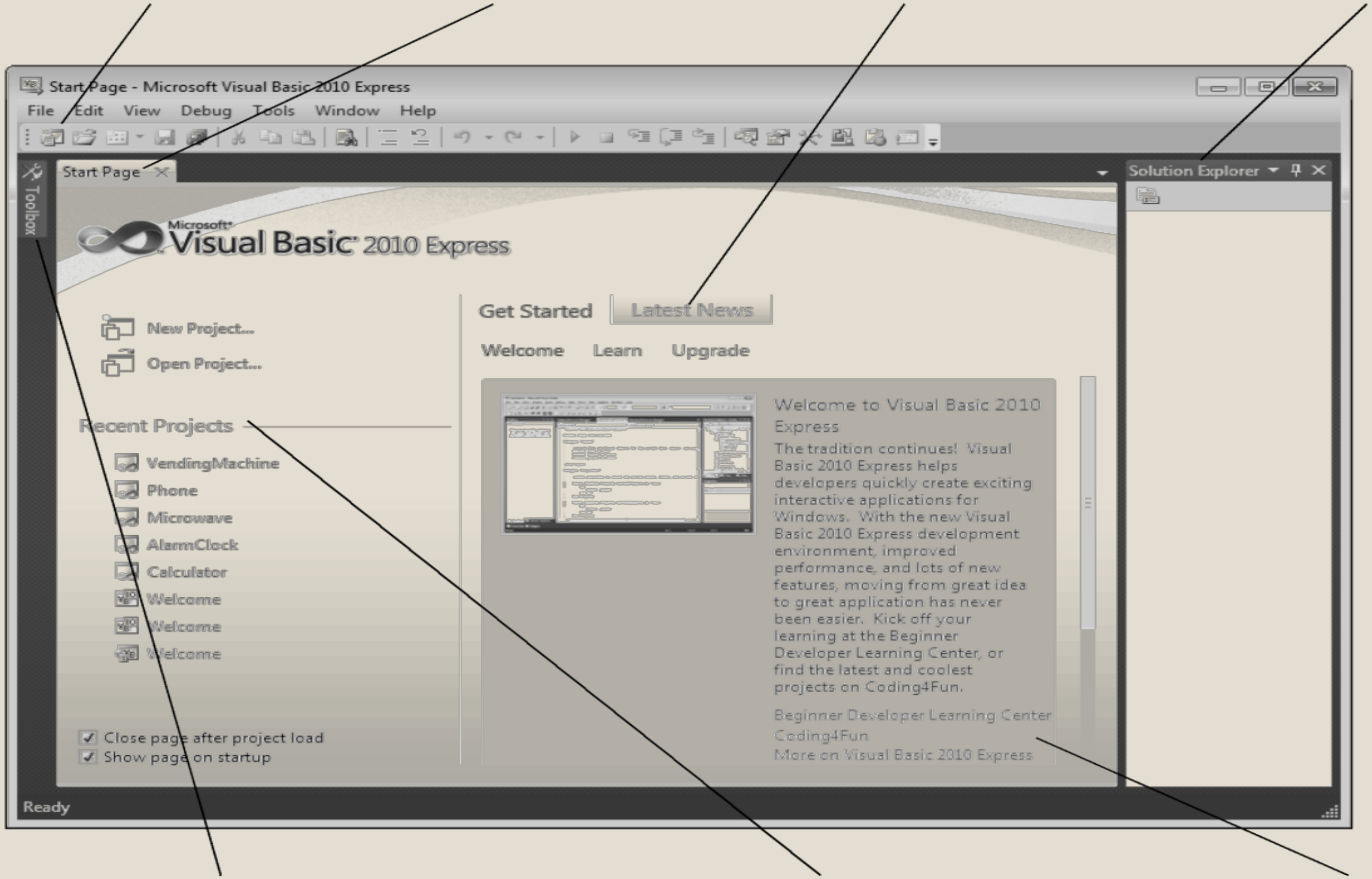
- Once the Express Edition begins execution, the **Start Page** displays (Fig. 2.3).
- The **Start Page** contains a list of links to Visual Basic 2010 IDE resources and web-based resources.

New Project button

Start Page tab

Latest News tab

Solution Explorer (no projects open)



Welcome to Visual Basic 2010 Express

The tradition continues! Visual Basic 2010 Express helps developers quickly create exciting interactive applications for Windows. With the new Visual Basic 2010 Express development environment, improved performance, and lots of new features, moving from great idea to great application has never been easier. Kick off your learning at the Beginner Developer Learning Center, or find the latest and coolest projects on Coding4Fun.

☑ Close page after project load

☑ Show page on startup

Collapsed **Toolbox** window

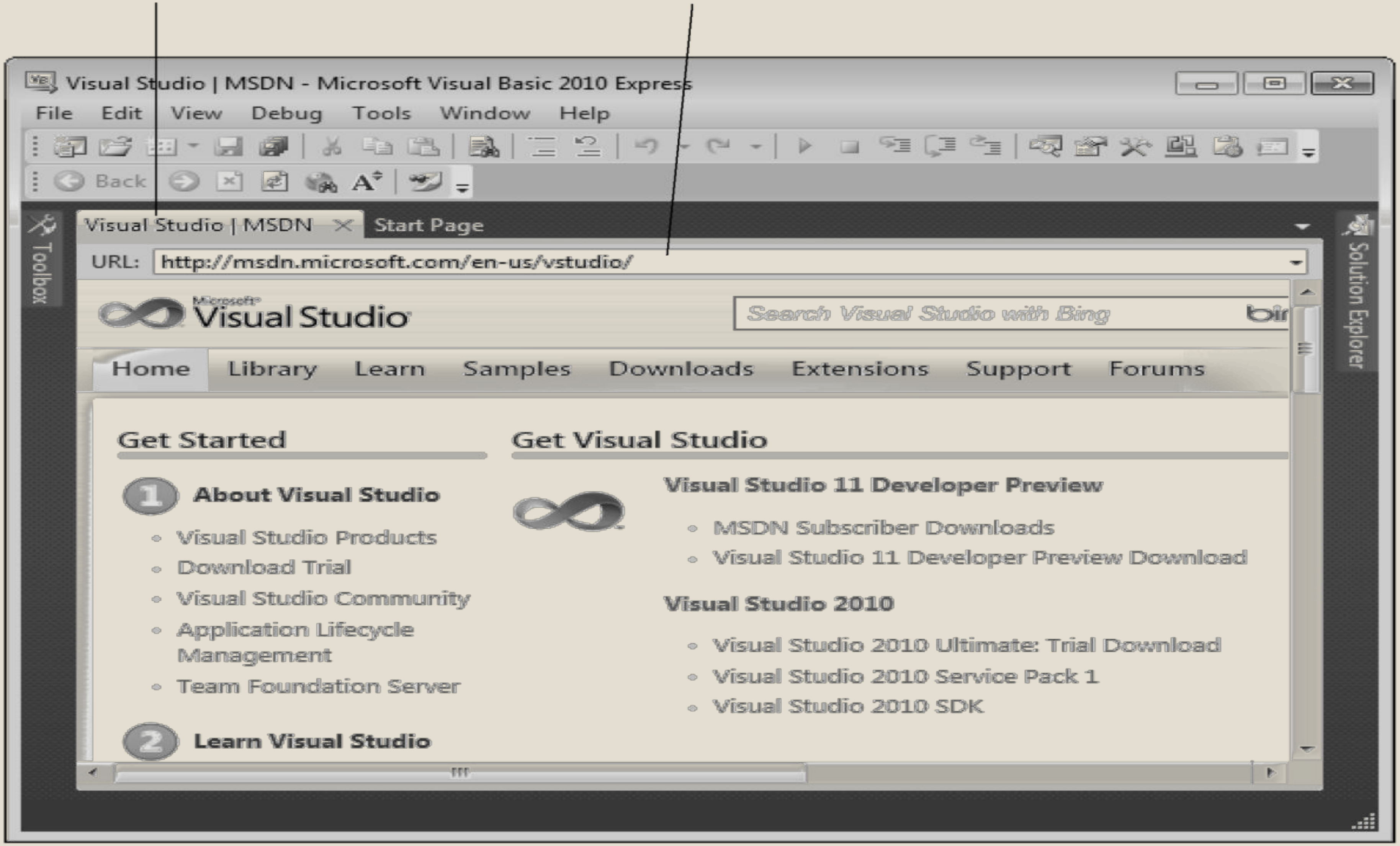
Recent projects will be listed here

Start Page links

Figure 2.3 **Start Page** in Visual Basic 2010 Express Edition.

2.2 Overview of the Visual Basic 2010 Express Edition IDE (Cont.)

- You can also browse the web from the IDE by selecting **View > Other Windows > Web Browser** or by typing **<Ctrl> <Alt> R**.
- To request a web page, type its URL into the **location *bar*** (Fig. 2.4) and press the *Enter* key—your computer, of course, must be connected to the Internet.
- The web page that you wish to view appears in the web browser tab in the IDE (Fig. 2.4).



Web browser window tab

Location bar

URL: http://msdn.microsoft.com/en-us/vstudio/

Microsoft® Visual Studio

Get Started

1 About Visual Studio

- Visual Studio Products
- Download Trial
- Visual Studio Community
- Application Lifecycle Management
- Team Foundation Server

2 Learn Visual Studio

Get Visual Studio

Visual Studio 11 Developer Preview

- MSDN Subscriber Downloads
- Visual Studio 11 Developer Preview Download

Visual Studio 2010

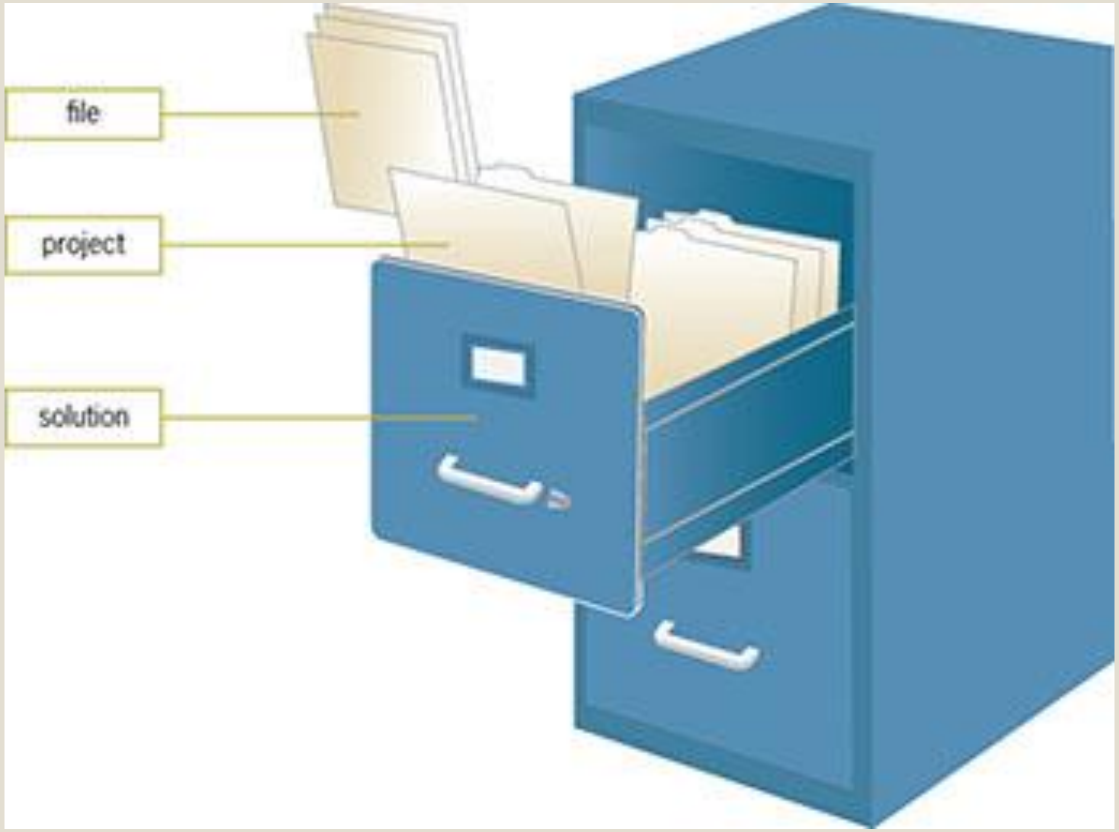
- Visual Studio 2010 Ultimate: Trial Download
- Visual Studio 2010 Service Pack 1
- Visual Studio 2010 SDK

Figure 2.4 Displaying a web page in Visual Studio.

2.3 Creating a Project for the Welcome App

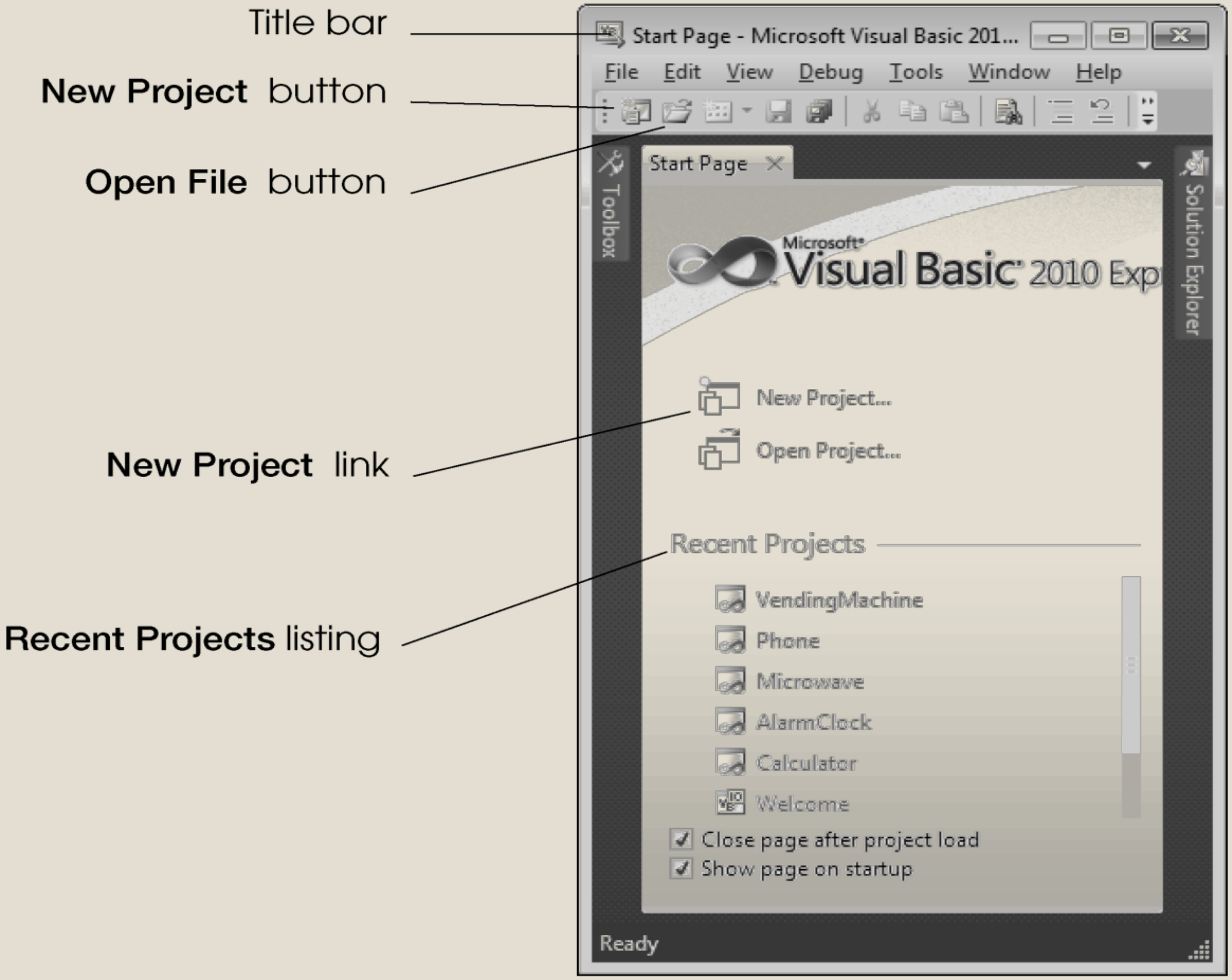
- Visual Basic organizes applications into **projects** and **solutions**.
- A project is a group of related files, such as the Visual Basic code and any images that might make up an app.
- A solution may contain one or more projects.

Solutions, Projects, and Files (cont'd.)



Creating a Project for the Welcome App

- There are several ways to create a new project or open an existing one:
  - Select either **File > New Project...** to create a new project or **File > Open Project...** to open an existing project.
  - From the **Start Page**, above the **Recent Projects** section, you can also click the links **New Project...** or **Open Project...**.
  - Click either the **New Project** `Button` (Fig. 2.5), causing the New Project dialog to display (Fig. 2.6), or the **Open File** `Button` (Fig. 2.5) which displays the **Open File** dialog.



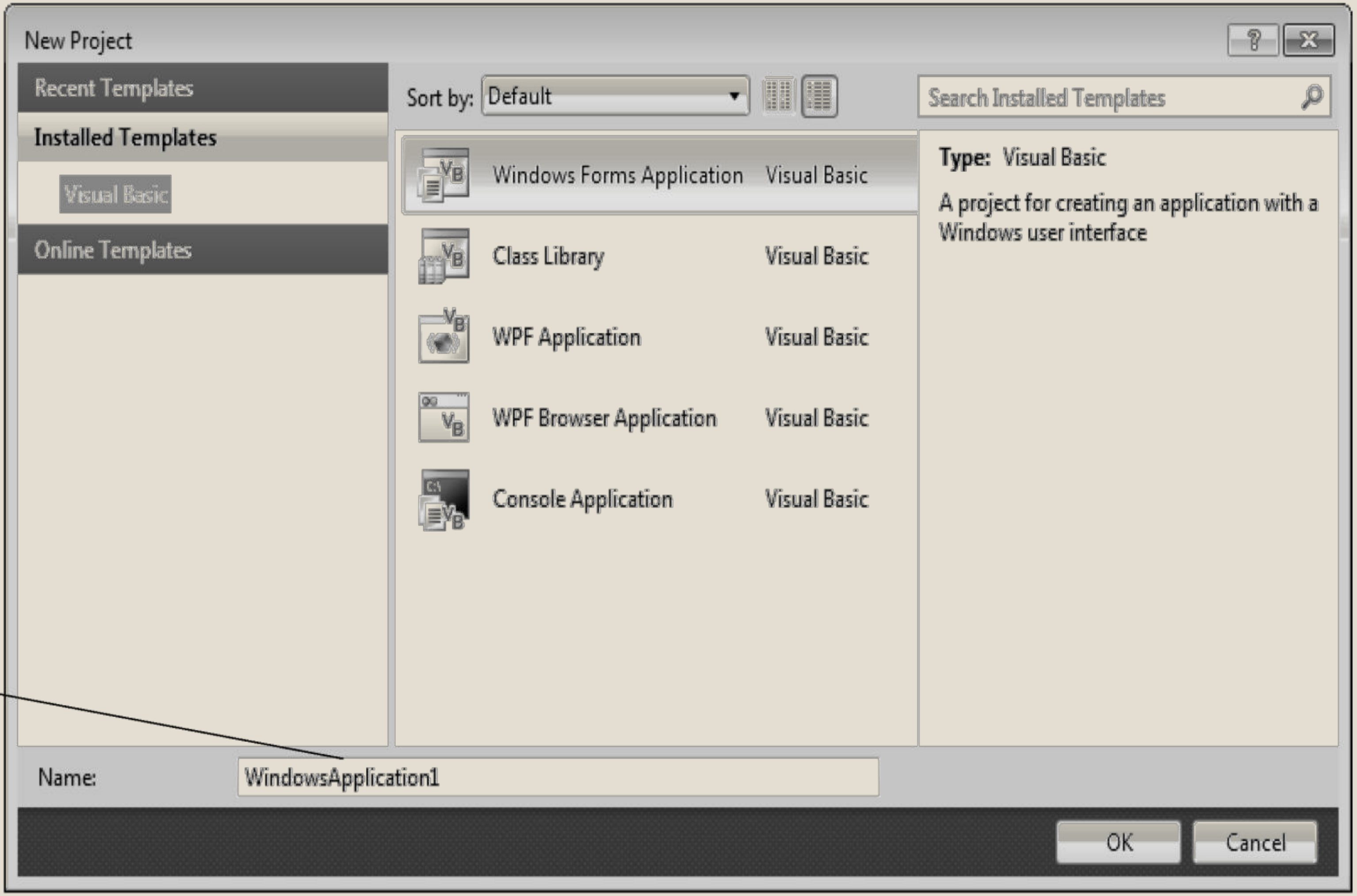
☑ Close page after project load
☑ Show page on startup

Figure 2.5 **New Project** button and **Recent Projects** listing.

Visual Studio 2010 (cont'd.)

- **Application**: program or suite of programs
- **Windows-based application**:
  - Has a Windows user interface
  - Runs on a personal computer
- **User interface**: what the user sees and interacts with when using an application
- **Web-based application**:
  - Has a Web user interface
  - Runs on a server
  - Accessed with a computer browser

- Visual Basic provides several templates (Fig. 2.6).
- **Templates** are the project types you can create in Visual Basic.
- Select **Windows Forms Application**.



New Project

Sort by: Default

Type: Visual Basic

A project for creating an application with a Windows user interface

Windows Forms Application	Visual Basic
Class Library	Visual Basic
WPF Application	Visual Basic
WPF Browser Application	Visual Basic
Console Application	Visual Basic

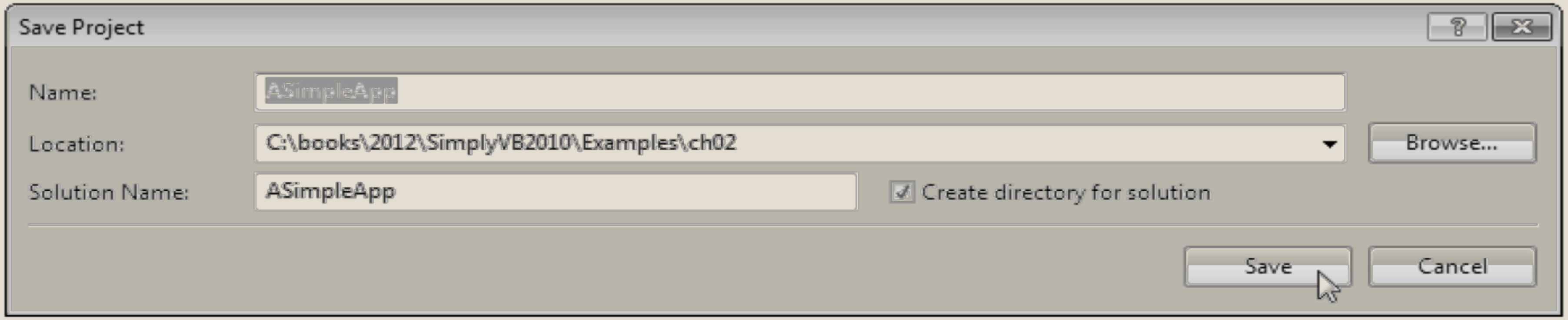
Name: WindowsApplication1

Default project name (provided by Visual Basic)

Figure 2.6 **New Project** dialog.

Creating a Project for the Welcome App (Cont.)

- To rename the project, type `ASimpleApp` in the **Name:** `TextBox`. Then click **OK**.
- Save this project in your `C:\SimplyVB2010` directory.
- If you want to change the project's location, select **File > Save All** (Fig. 2.7).



Name: ASimpleApp

Location: C:\books\2012\SimplyVB2010\Examples\ch02

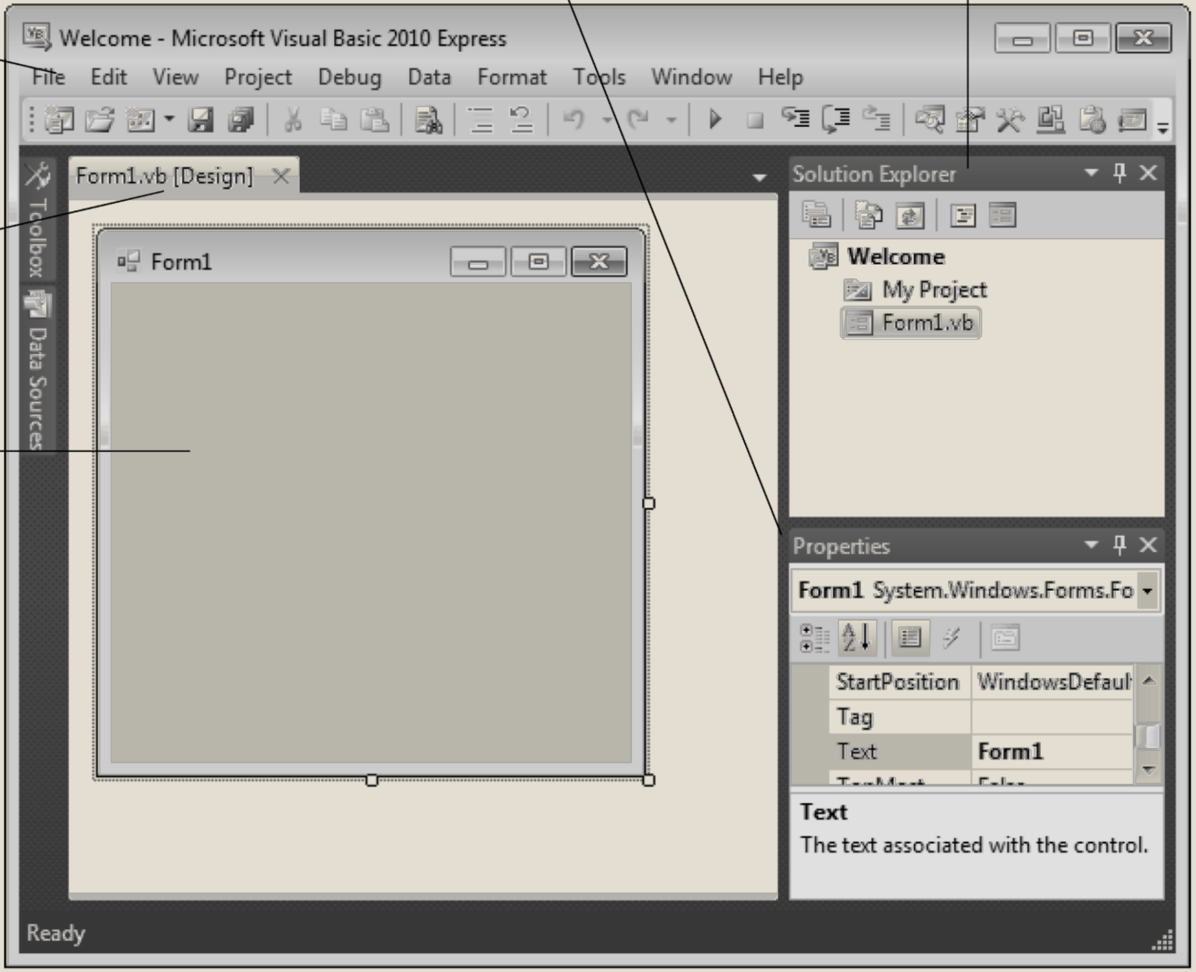
Solution Name: ASimpleApp

☑ Create directory for solution

Figure 2.7 **Save Project** dialog.

Creating a Project for the Welcome App (Cont.)

- This displays the IDE in **Design** view (Fig. 2.9), which enables you to design GUIs.
- The IDE's **Design** view is also known as the **Windows `Forms` Designer**.



StartPosition	WindowsDefaul
Tag	
Text	**Form1**

Text
The text associated with the control.

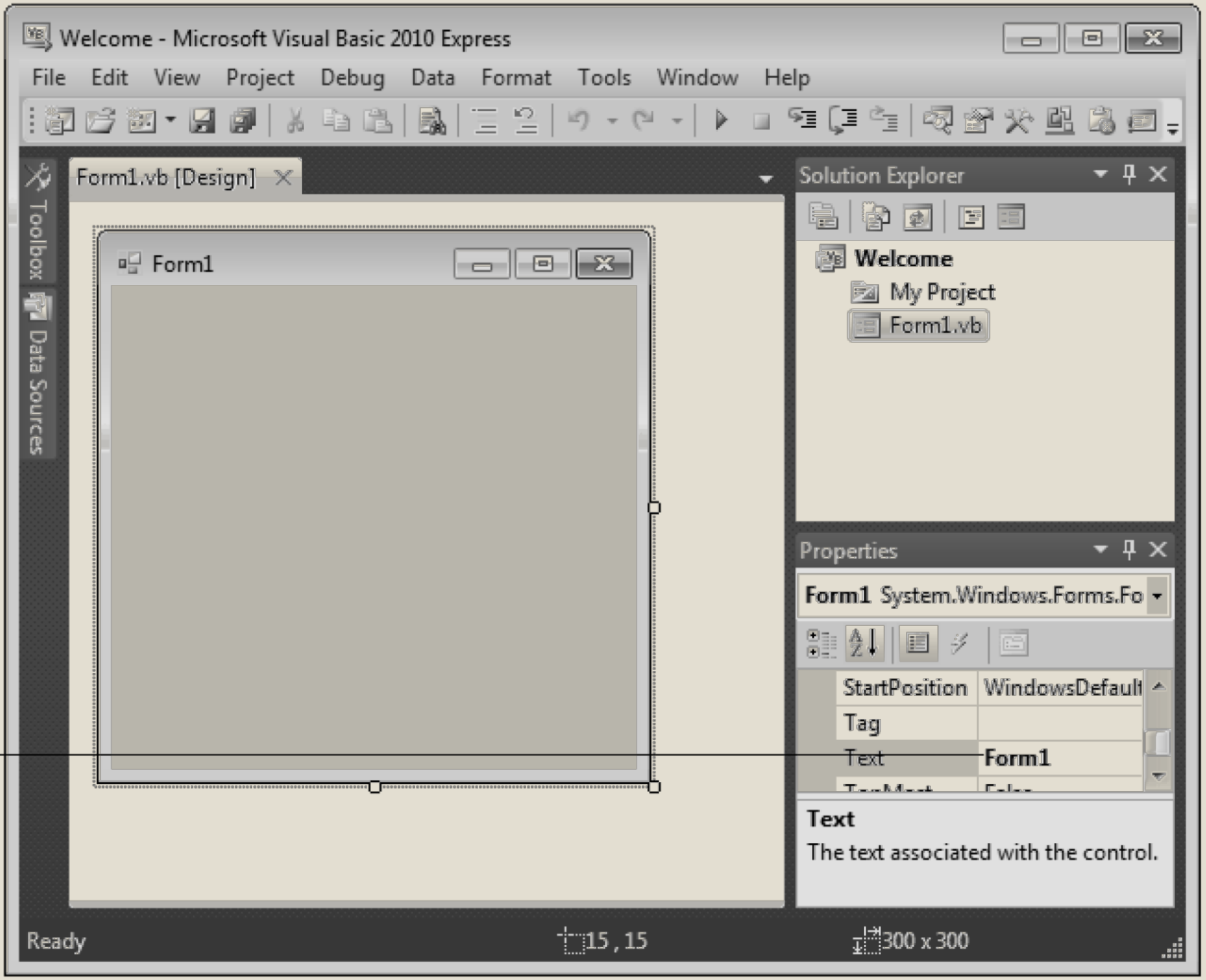
Figure 2.9 **Design** view of the IDE.

Creating a Project for the Welcome App (Cont.)

- The rectangle in the **Design** area titled **`Form1`** (called a **`Form`**) represents the main window of the app you're creating.
- Visual Basic apps can have multiple `Forms`—however, most apps you'll create will use only one.
- You'll learn how to customize the `Form` by adding GUI **controls**—in this example, you'll add a `Label` and a `PictureBox`.

Creating a Project for the Welcome App (Cont.)

- A **`Label`** typically contains descriptive text (for example, `"Welcome to Visual Basic!"`), and a **`PictureBox`** displays an image, (select your preferable Image).
- Visual Basic Express has many *preexisting* controls and other components you can use to build and customize your apps. Many of these controls are discussed and used throughout this course.
- Other controls are available from third parties.



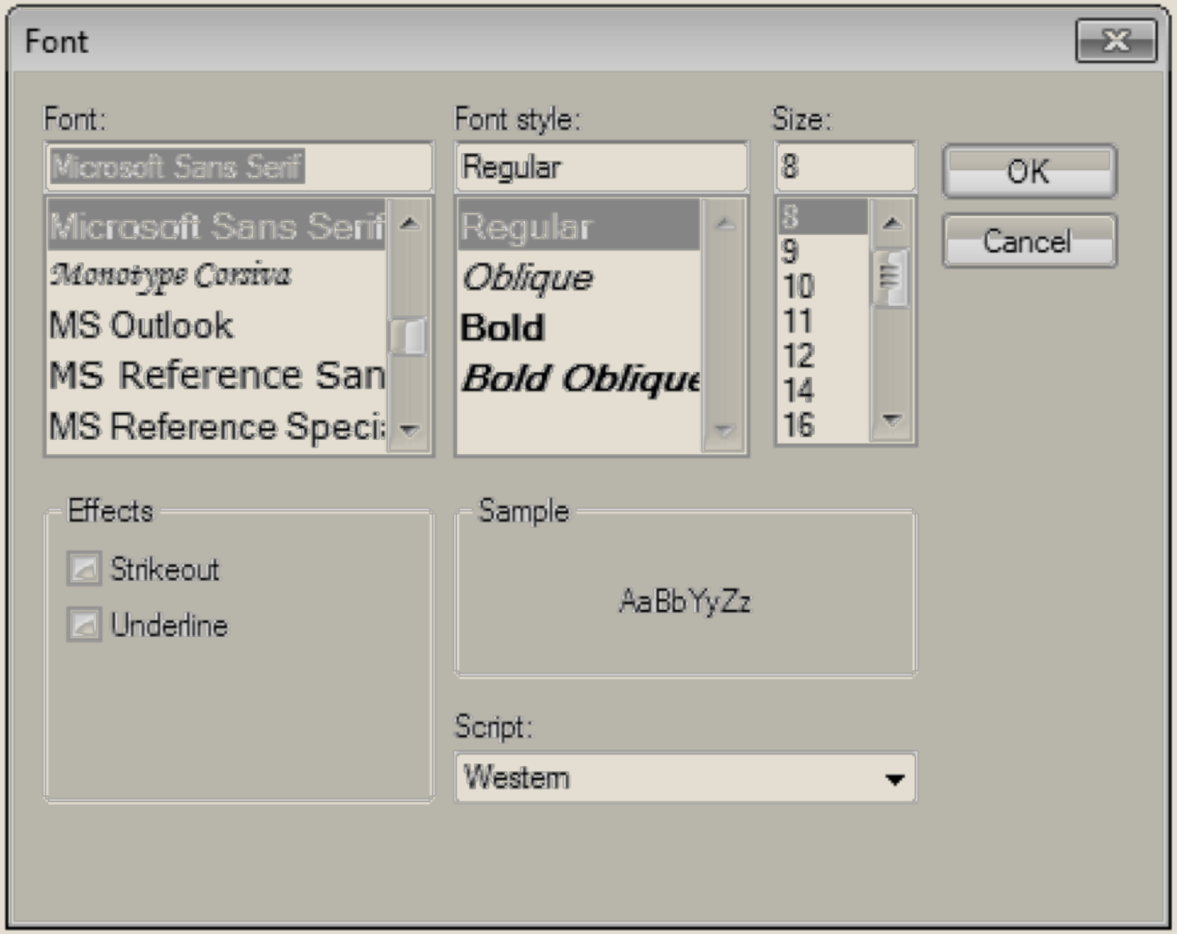
Form1 System.Windows.Forms.Fo

StartPosition	WindowsDefault
Tag	
Text	Form1

Text
The text associated with the control.

Textbox (displaying the text `Form1`) which can be modified

Figure 2.10 Textbox control for modifying a property in the Visual Studio IDE.



Font

Font: Microsoft Sans Serif

Font style: Regular

Size: 8

Effects

☐ Strikeout

☐ Underline

Sample

AaBbYyZz

Script: Western

Figure 2.11 Dialog for modifying a control's font properties.

2.4 Menu Bar and Toolbar

- Commands for managing the IDE and for developing, maintaining and executing apps are contained in **menus**, which are located on the **menu bar** of the IDE (Fig. 2.12).

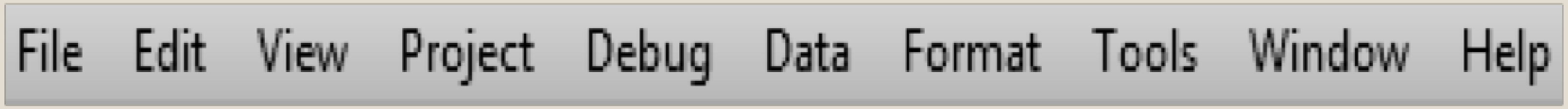


Figure 2.12 Visual Studio menu bar.

2.4 Menu Bar and Toolbar (Cont.)

- Menus contain groups of related commands (also called **menu items**) that, when selected ,cause the IDE to perform specific actions (for example, open a window, save a file, print a file and execute an app).

Menu	Description
File	Contains commands for opening, closing, adding and saving projects, as well as printing project data and exiting Visual Studio.
Edit	Contains commands for editing programs, such as cut, copy, paste, undo, redo, delete, find and select.
View	Contains commands for displaying IDE windows (for example, **Solution Explorer**, **Toolbox**, **Properties** window) and for adding toolbars to the IDE.
Project	Contains commands for managing projects and their files.
Debug	Contains commands for compiling, debugging (that is, identifying and correcting problems in apps) and running apps.
Data	Contains commands for interacting with databases (that is, organized collections of data stored on computers), which we discuss in Chapter 27.
Format	Contains commands for arranging and modifying a `Form`'s controls. The **Format** menu appears only when a GUI component is selected in **Design** view.
Tools	Contains commands for accessing additional IDE tools and options for customizing the IDE.

Figure 2.13 Summary of Visual Studio 2010 IDE menus. (Part 1 of 2.)

2.4 Menu Bar and Toolbar (Cont.)

- You can access many common commands from the IDE **toolbar** (Fig. 2.14), which contains graphics, called **icons**, that graphically represent commands.
- By default, the standard toolbar is displayed when you run Visual Studio for the first time—it contains icons for the most commonly used commands, such as opening a file, adding an item to a project, saving files and running apps (Fig. 2.14).

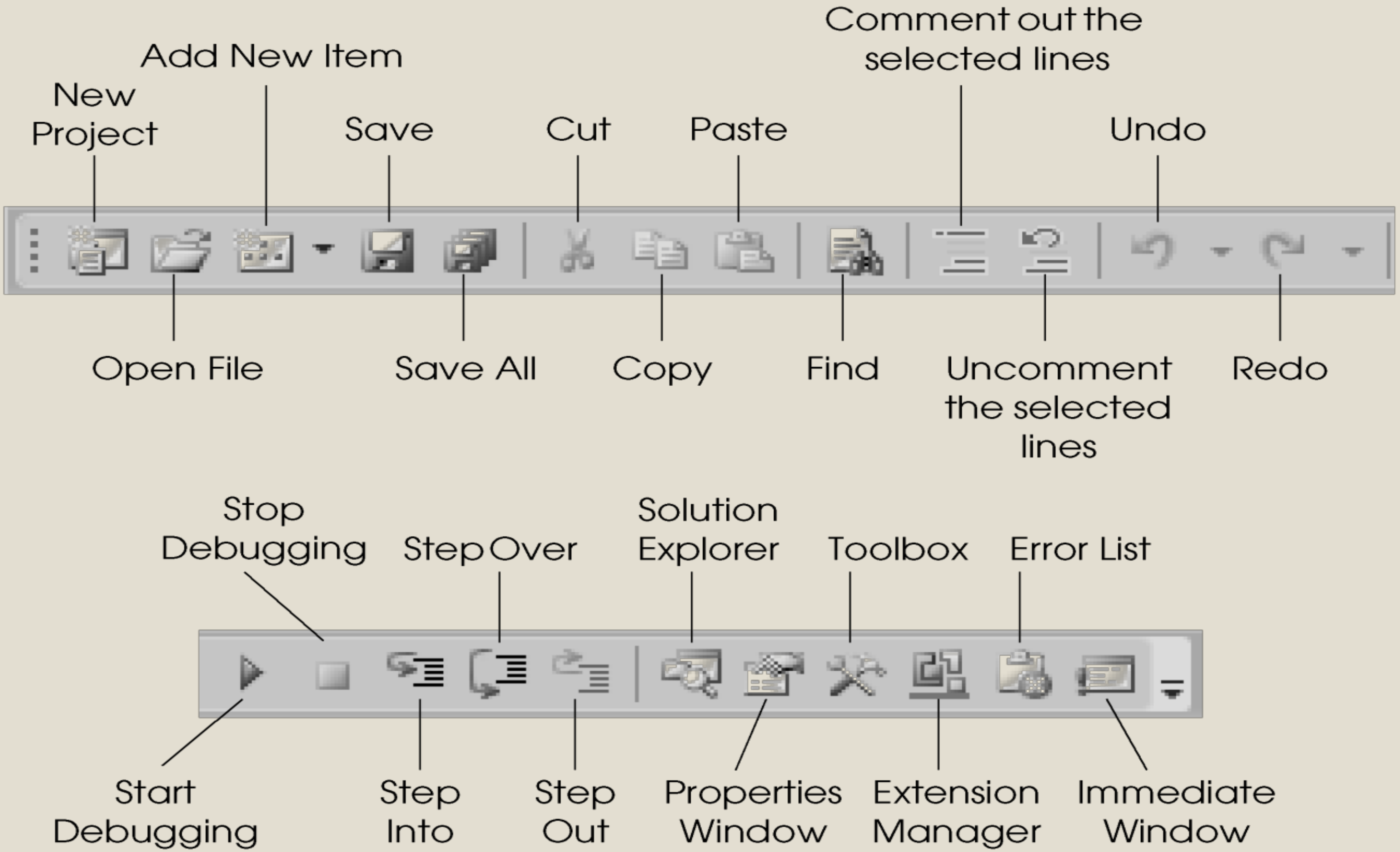
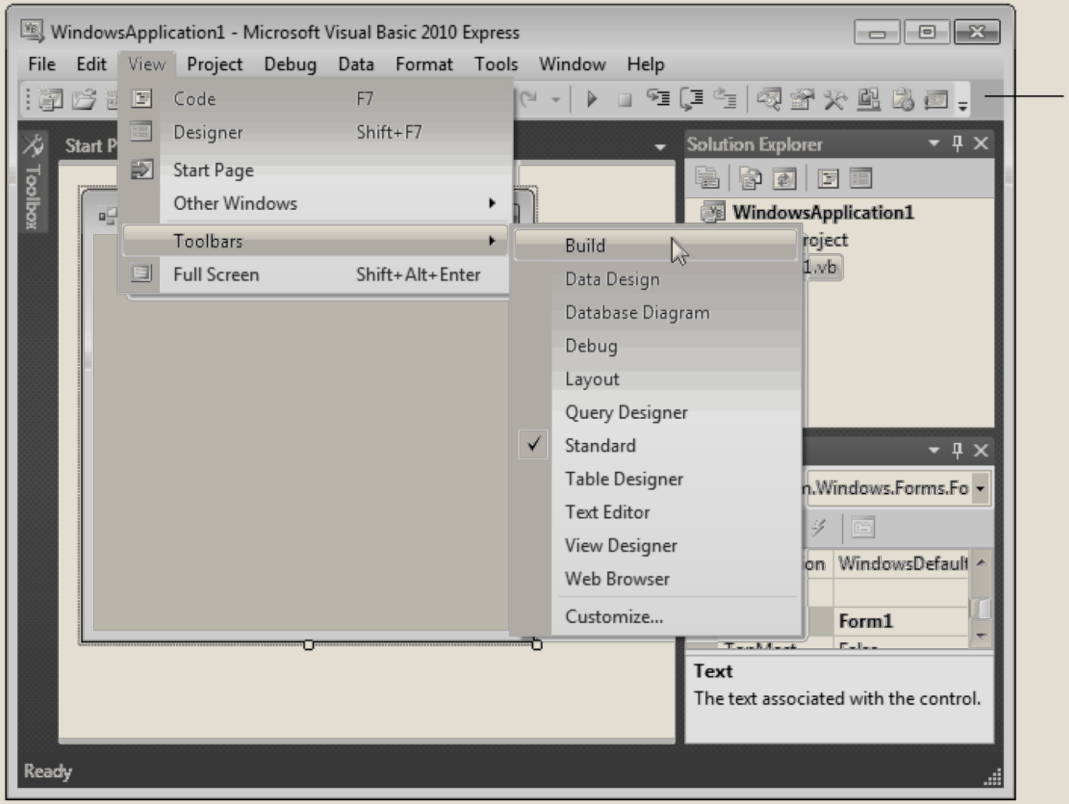


Figure 2.14 Standard Visual Studio toolbar.

2.4 Menu Bar and Toolbar (Cont.)

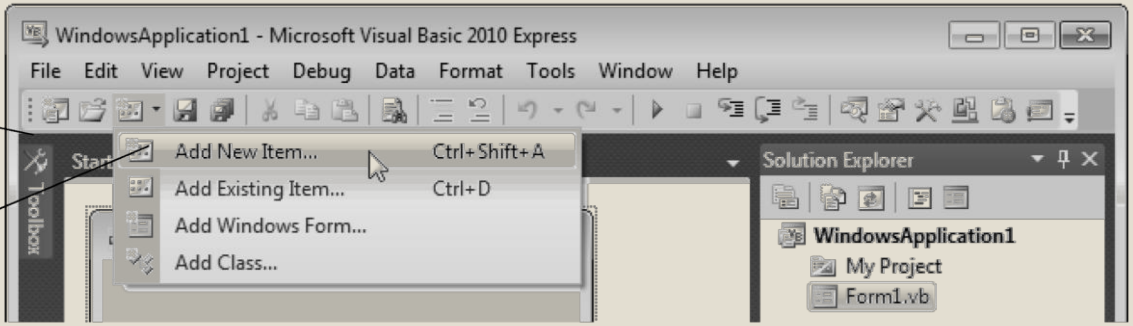
- You can customize the IDE’s toolbars.
- Select **View > Toolbars** (Fig. 2.15).



Text
The text associated with the control.

Additional toolbars appear to the right of or below other toolbars

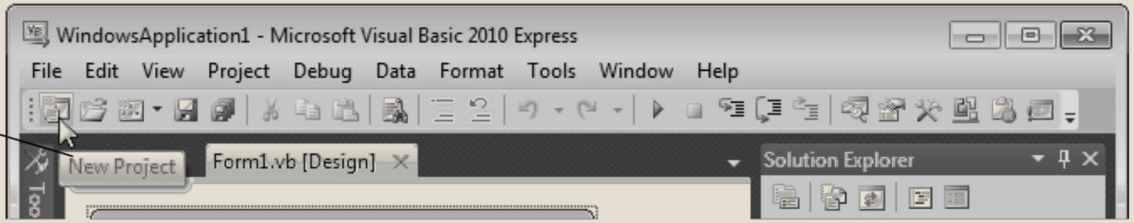
Figure 2.15 Adding the **Build** toolbar to the IDE.


WindowsApplication1 - Microsoft Visual Basic 2010 Express
File Edit View Project Debug Data Format Tools Window Help
Add New Item... Ctrl+Shift+A
Add Existing Item... Ctrl+D
Add Windows Form...
Add Class...
Solution Explorer
WindowsApplication1
My Project
Form1.vb
Toolbox

Toolbar

Down arrow indicates additional commands

Figure 2.16 IDE toolbar icon showing additional commands.

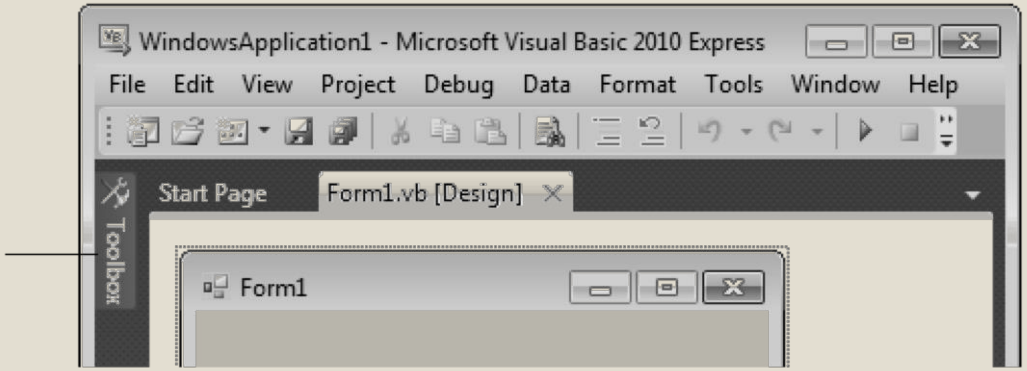

WindowsApplication1 - Microsoft Visual Basic 2010 Express
File Edit View Project Debug Data Format Tools Window Help
New Project
Form1.vb [Design]
Solution Explorer

Tool tip appears when the mouse pointer rests on an icon for a few seconds

Figure 2.17 Tool tip demonstration.

2.5 Navigating the Visual Studio IDE; Auto-Hide (Cont.)

- Visual Studio provides a space-saving feature called **auto-hide**.
- When auto-hide is enabled, a tab appears along either the left, right or bottom edge of the IDE window (Fig. 2.19).
- This tab contains one or more icons, each of which identifies a hidden window.
  - Placing the mouse pointer over one of these icons displays that window (Fig. 2.20).
  - Moving the mouse pointer outside the window's area hides the window.
- To "pin down" a window (that is, to disable auto-hide and keep the window open), click the pin icon.
  - When auto-hide is enabled, the pin icon is horizontal (Fig. 2.20)—when a window is "pinned down," the pin icon is vertical (Fig. 2.21).



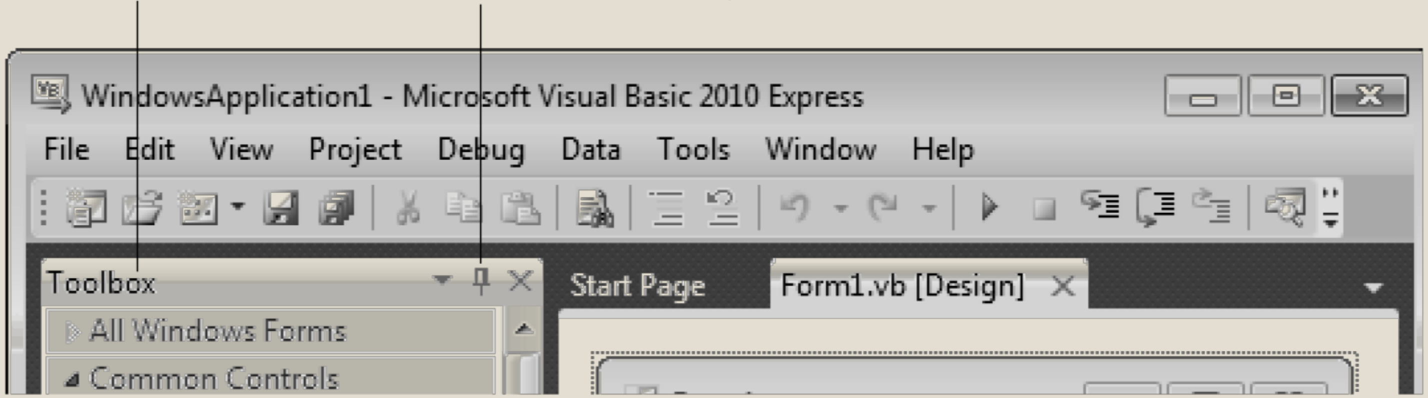
Icon for hidden window (auto-hide enabled)

WindowsApplication1 - Microsoft Visual Basic 2010 Express
File Edit View Project Debug Data Format Tools Window Help
Toolbox
Start Page
Form1.vb [Design]
Form1

Figure 2.19 Auto-hide feature demonstration.

Toolbox "pinned down"

Vertical orientation for pin icon when window is "pinned down"



WindowsApplication1 - Microsoft Visual Basic 2010 Express
File Edit View Project Debug Data Tools Window Help
Toolbox
All Windows Forms
Common Controls
Start Page
Form1.vb [Design]

Figure 2.21 Disabling auto-hide ("pinning down" a window).

2.5 Navigating the Visual Studio IDE; Auto-Hide (Cont.)

Solution Explorer

- The **Solution Explorer** window (Fig. 2.22) provides access to all of a solution’s files.
- If the window is not shown in the IDE, click the **Solution Explorer** icon in the IDE (Fig. 2.18), select **View > Other Windows >Solution Explorer** or type *<Ctrl> <Alt> L.*
- Click the **Show All Files** icon (Fig. 2.22) to display all the files in the solution.
- For your single-project solution, **Welcome** is the only project. The `Form` file is named `Form1.vb`.

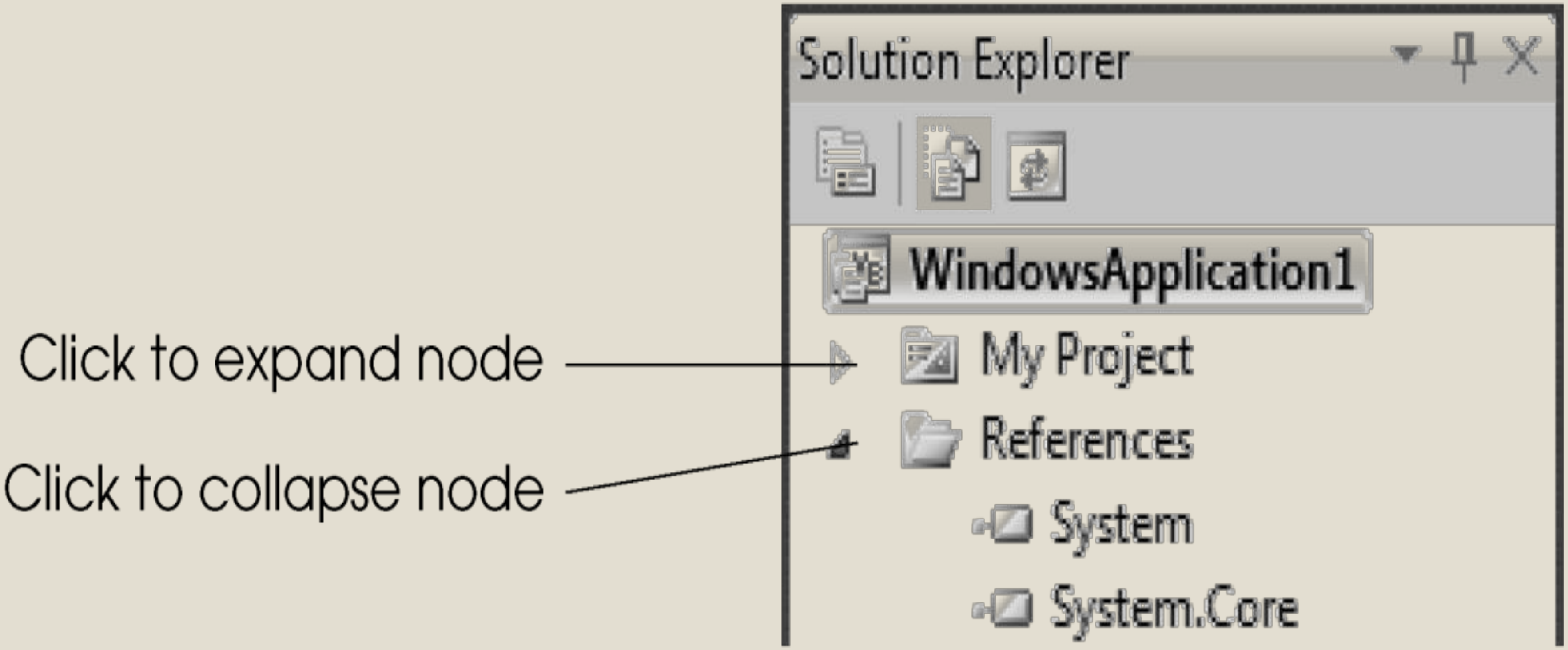


Figure 2.23 Expanded node.

2.5 Navigating the Visual Studio IDE; Auto-Hide (Cont.)

Toolbox

- The **Toolbox** (**View > Other Windows > Toolbox**) contains icons representing controls used to customize `Forms` (Fig. 2.24).
- With visual app development, you can "drag and drop" controls onto the Form and the IDE will write the code that creates the controls for you.
- Reusing preexisting controls saves time and money when you develop apps.

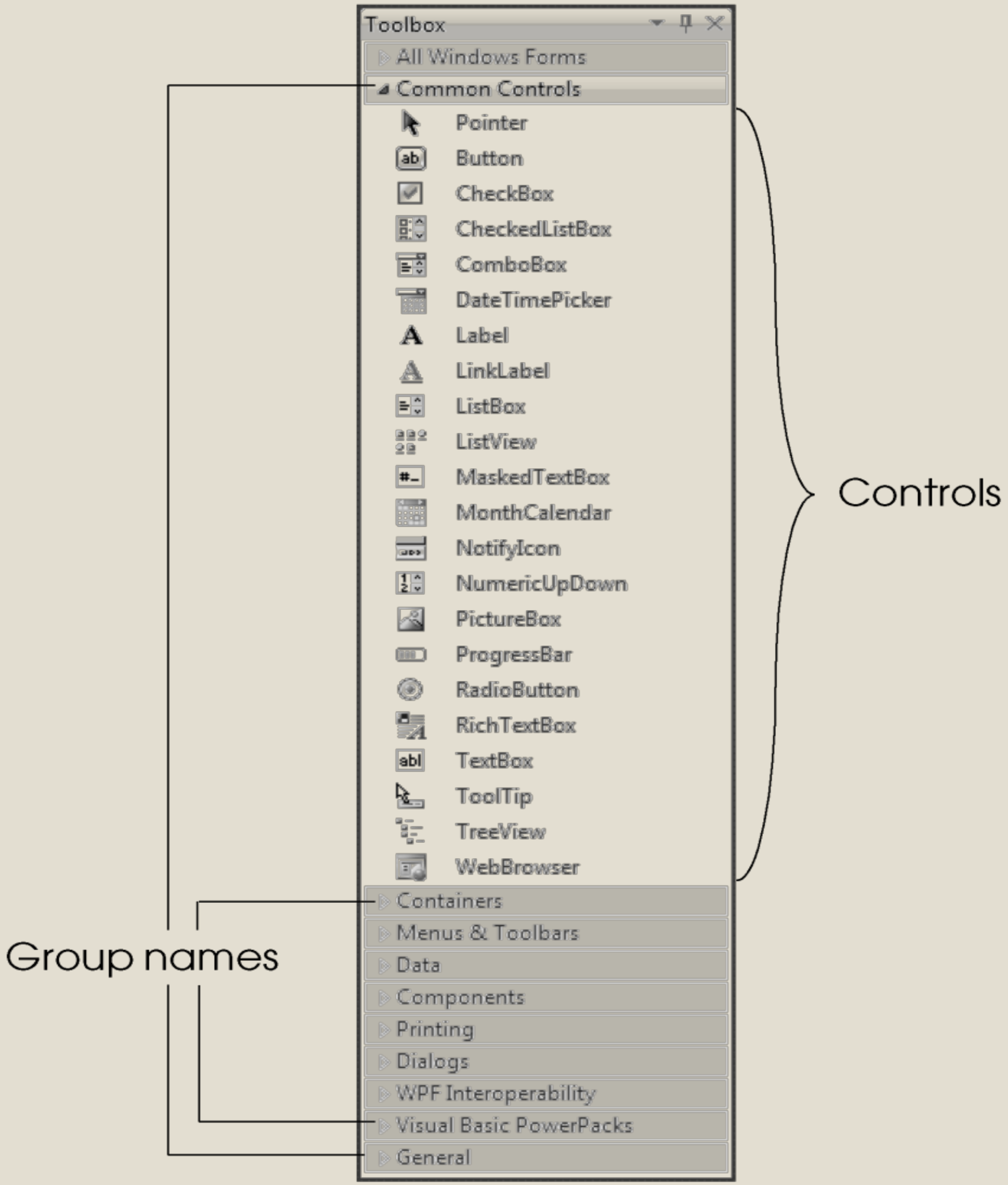
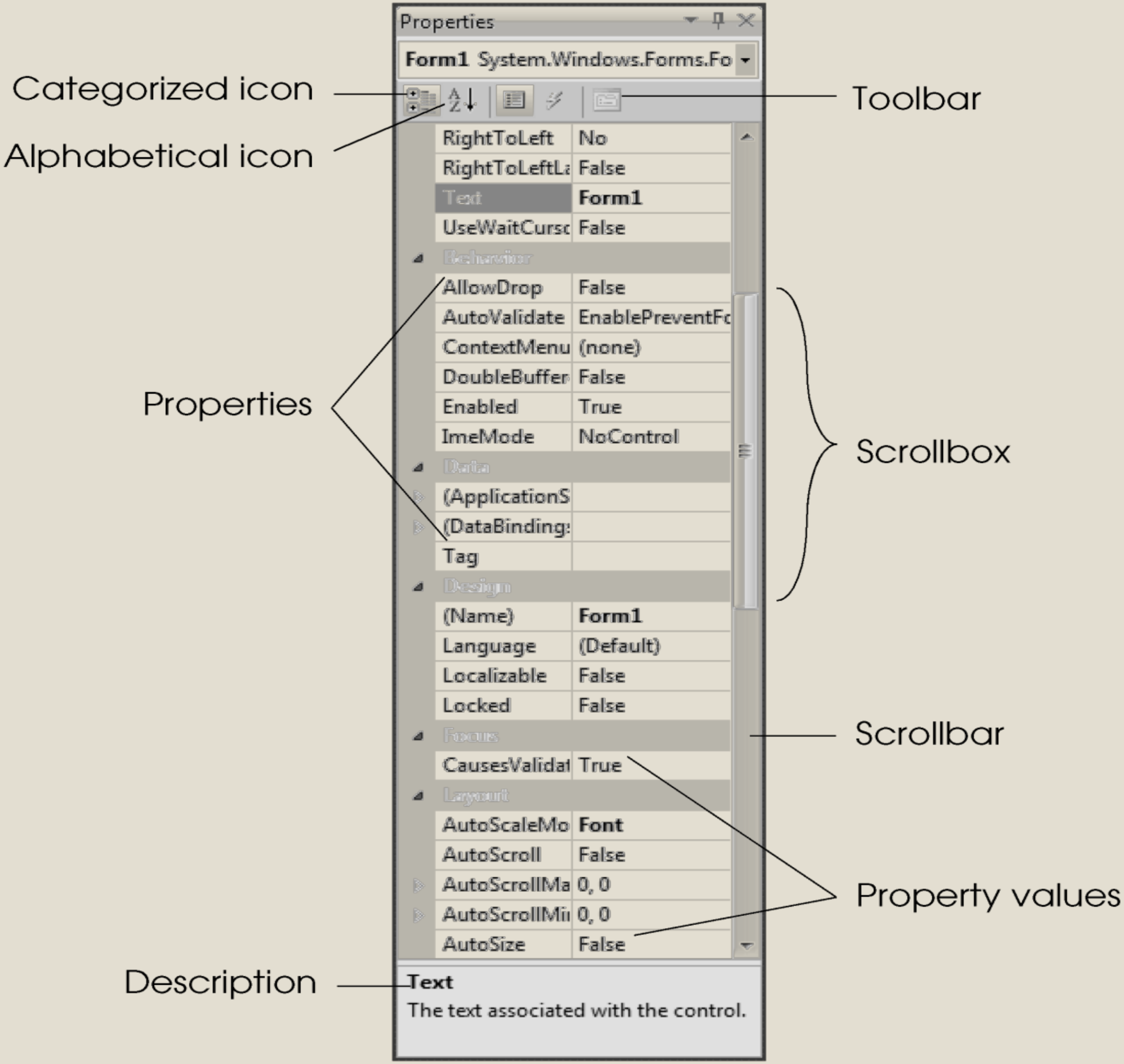


Figure 2.24 **Toolbox** window displaying controls for the **Common Controls** group.

2.5 Navigating the Visual Studio IDE; (Cont.)

Properties Window

- To display the **Properties** window, select **View > Other Windows > Properties** Window, click the **Properties** window toolbar icon shown in Fig. 2.18, or press the *F4* key.
- The **Properties** window displays the properties for the currently selected Form (Fig. 2.25), control or file in **Design** view.
- **Properties** specify information about the `Form` or control, such as its size, color and position.



Properties

Form1 System.Windows.Forms.Fo

RightToLeft	No
RightToLeftLa	False
Text	**Form1**
UseWaitCursc	False
Behavior	
AllowDrop	False
AutoValidate	EnablePreventFc
ContextMenu	(none)
DoubleBuffer	False
Enabled	True
ImeMode	NoControl
Data	
(ApplicationS	
(DataBinding:	
Tag	
Design	
(Name)	**Form1**
Language	(Default)
Localizable	False
Locked	False
Focus	
CausesValidat	True
Layout	
AutoScaleMo	**Font**
AutoScroll	False
AutoScrollMa	0, 0
AutoScrollMi	0, 0
AutoSize	False

Text
The text associated with the control.

Figure 2.25 Properties window.

Navigating a Project with the Solution Explorer (Cont.)

- Figure 2.25 shows `Form1`'s **Properties** window.
- The left column lists the `Form`'s properties—the right column displays the current value of each property.
- You can sort the properties either alphabetically (by clicking the **Alphabetical icon**) or categorically (by clicking the **Categorized icon**) according to their use (that is, **Appearance**, **Behavior**, **Design**, etc.).
- Depending on the size of the Properties window, some of the properties may be hidden from view on the screen.
- Users can scroll through the list of properties by **dragging** the **scrollbox** up or down inside the **scrollbar**, or by clicking the arrows at the top and bottom of the scrollbar.

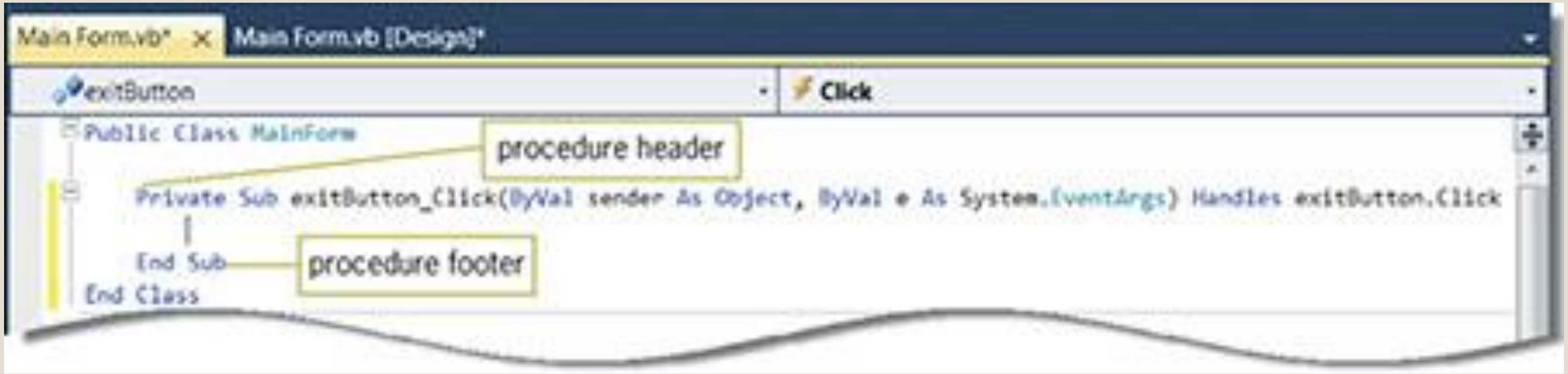
The Label Control

- **Label control**:
  - Displays text that user cannot edit
  - Used as “prompts” to explain controls or display output
  - Name should end with “Label”
- Control names use camel case
- **Camel case**: lowercase first word; uppercase first letter of each subsequent word in the name
- Not necessary to assign meaningful names for labels used as prompts because they are never used in code
- Labels used for output should have meaningful names

The Picture Box Control

- **Picture box control**: used to display an image on a form
- **Image** property: specifies the image to display
- **SizeMode** property: handles how the image will be displayed
  - Settings: Normal, StretchImage, AutoSize, CenterImage, or Zoom

The Code Editor Window



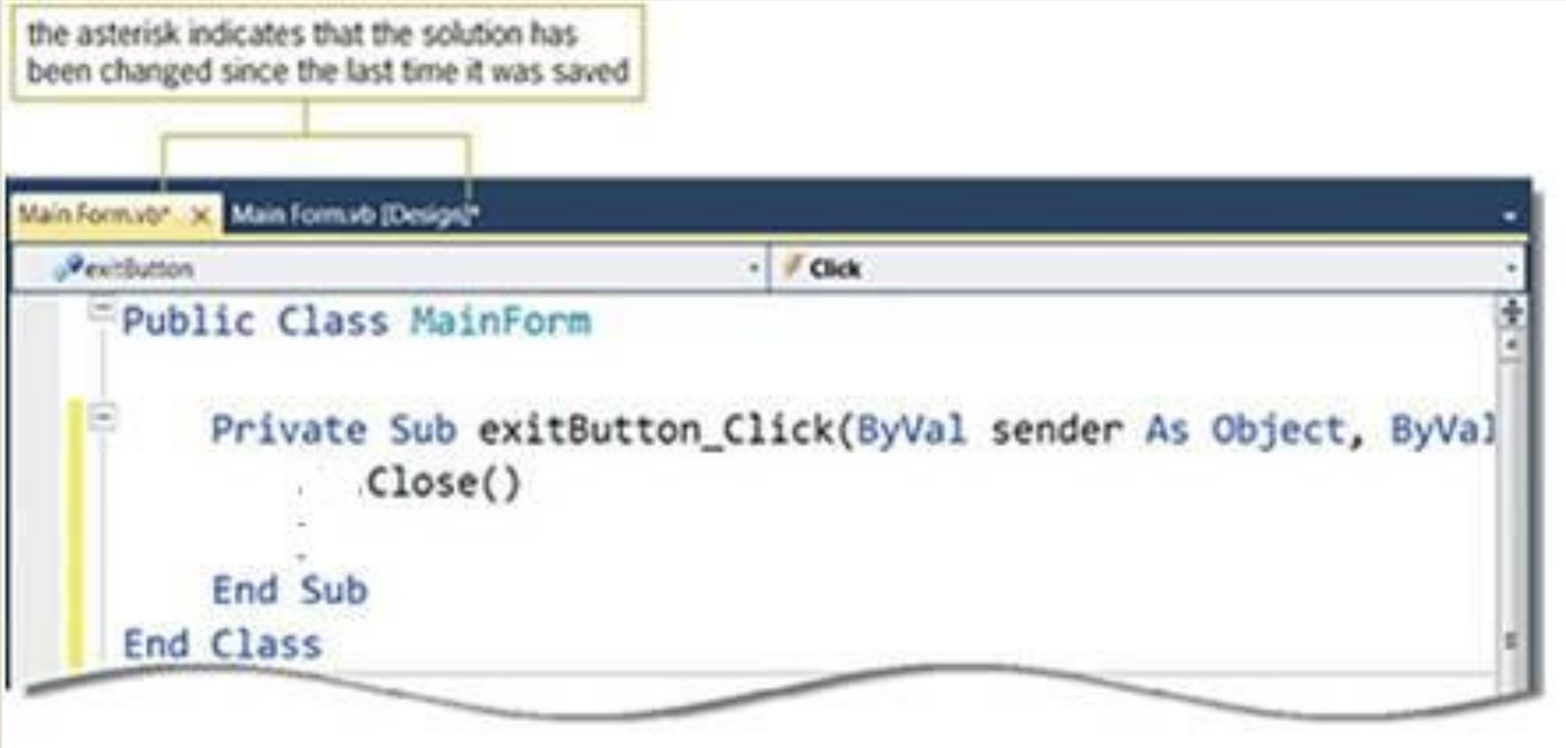
```
Public Class MainForm
    Private Sub exitButton_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles exitButton.Click

    End Sub
End Class
```

Figure 21: Code template for the exitButton’s Click event procedure

The Close() Instruction

- `Close()` **instruction**: closes the current form at run time
  - If the current form is the only form, the application is terminated
- **Method**: predefined VB procedure that can be invoked (called) when needed

The Close() Instruction (cont'd.)



the asterisk indicates that the solution has been changed since the last time it was saved

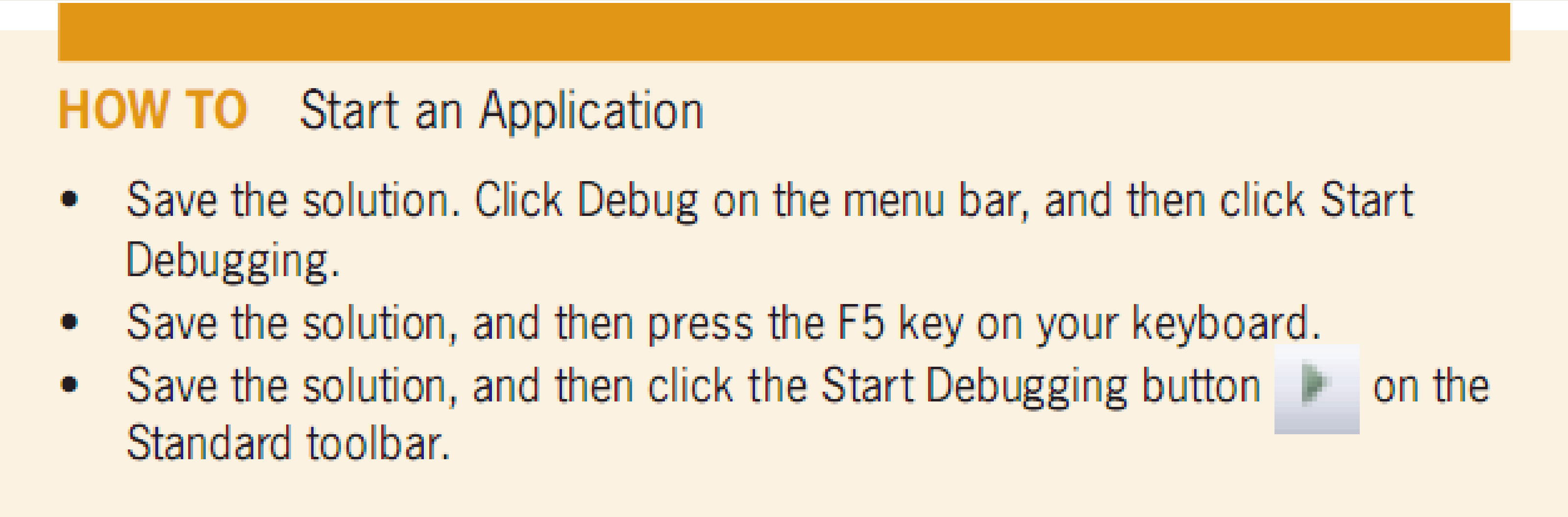
```
Public Class MainForm

    Private Sub exitButton_Click(ByVal sender As Object, ByVal
        Close()

    End Sub
End Class
```

The `Close()` instruction entered in the Click event procedure

Navigating a Project with the Solution Explorer (Cont.)

- The **Properties** window is crucial to visual app development—it allows you to modify a control's properties visually, *without writing code*.
- At the top of the **Properties** window is the **component selection drop-down list**, which allows you to select the `Form` or control to display its properties in the **Properties** window (Fig. 2.25).

Starting and Ending an Application (cont'd.)



HOW TO Start an Application

- Save the solution. Click Debug on the menu bar, and then click Start Debugging.
- Save the solution, and then press the F5 key on your keyboard.
- Save the solution, and then click the Start Debugging button ▶ on the Standard toolbar.

How to start an application

2.6 Using Help

■ Microsoft provides extensive help documentation via the **Help menu**.

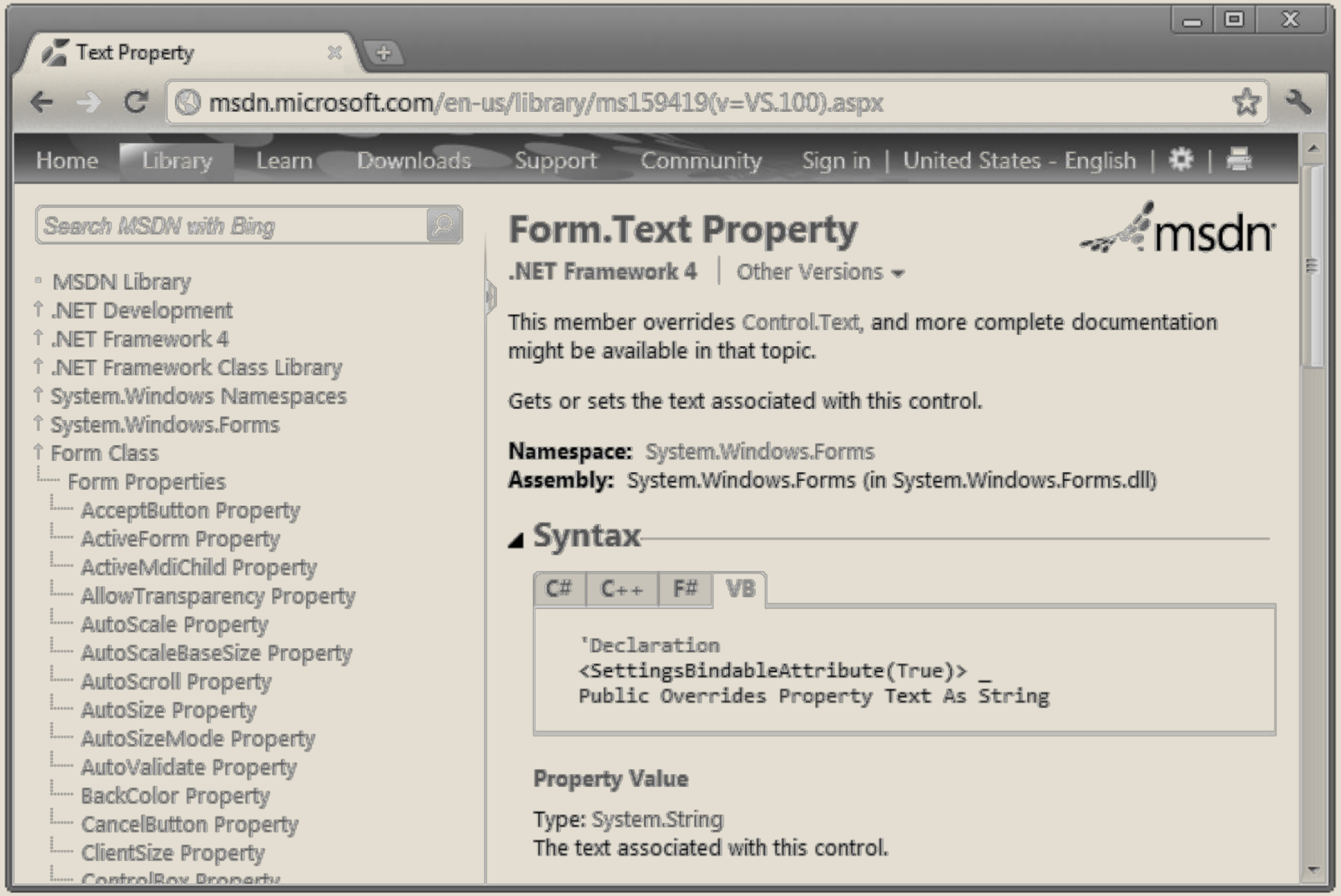
■ Using **Help** is an excellent way to get information quickly about Visual Studio, Visual Basic and more.

Context-Sensitive Help

■ Visual Studio provides **context-sensitive help** pertaining to the “current content” (that is, the items around the location of the mouse cursor).

2.6 Using Help (Cont.)

- To use context-sensitive help, click an item, such as the `Form`, then press the *F1* key.
- The help documentation is displayed in a web browser window.
- To return to the IDE, either close the browser window or select the icon for the IDE in your Windows task bar.
- Figure 2.26 shows the help page for a `Form`’s `Text` property.



Form.Text Property

This member overrides Control.Text, and more complete documentation might be available in that topic.

Gets or sets the text associated with this control.

Namespace: System.Windows.Forms
Assembly: System.Windows.Forms (in System.Windows.Forms.dll)

Syntax

```
'Declaration
<SettingsBindableAttribute(True)> _
Public Overrides Property Text As String
```

Property Value

Type: System.String
The text associated with this control.

Figure 2.26 Using context-sensitive help.

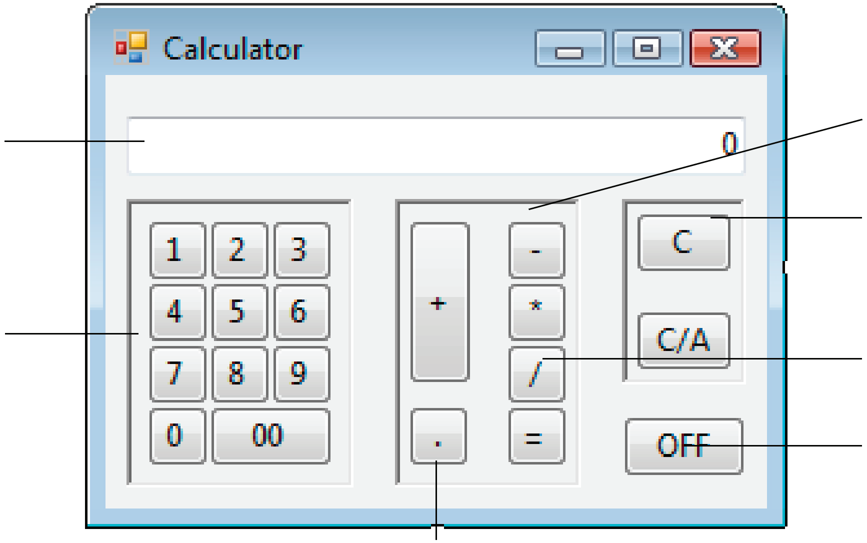
2.7 Saving and Closing Projects in Visual Basic

Once you're finished with a project, you should save the project's files and close the project.

- Before closing the project for the **Welcome** app, you should save its files, ensuring that any changes made to the project are not lost.
- Although you did not make any changes, save your project files by selecting **File > Save All**.
- Select **File > Close Project**.

Homework

*(Calculator **GUI**)* Create the GUI for the calculator shown in the figure



TextBox

Panel (contains 11 Buttons for the numeric keys)

Subtraction Button

Panel (contains 2 Buttons)

Panel (contains 6 Buttons)

Button

Decimal point Button