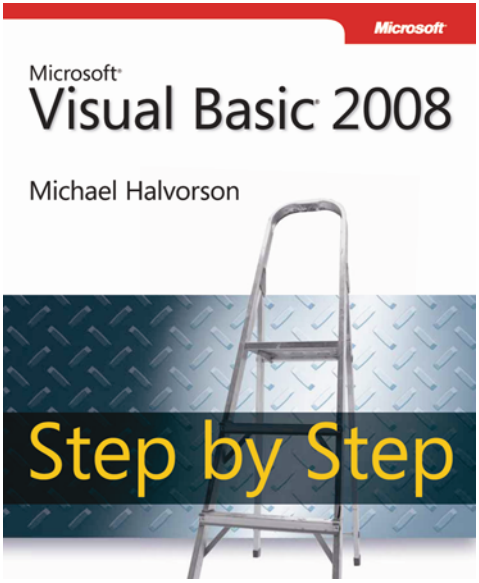


Microsoft® Visual Basic® 2008 Step by Step

Michael Halvorson

To learn more about this book, visit Microsoft Learning at
http://www.microsoft.com/MSPress/books/12202.aspx

9780735625372

Microsoft® Press

Table of Contents

Part IV Database and Web Programming

Appendix

What do you think of this book? We want to hear from you!

Microsoft is interested in hearing your feedback so we can continually improve our books and learning resources for you. To participate in a brief online survey, please visit:

www.microsoft.com/learning/booksurvey/

Chapter 20

Creating Web Sites and Web Pages by Using Visual Web Developer and ASP.NET

After completing this chapter, you will be able to:

- Start Visual Web Developer and create a new Web site.
- Use Visual Web Developer tools and windows, including the Web Page Designer.
- Use the Visual Web Developer Toolbox to add server controls to Web pages.
- Add text, formatting effects, and Visual Basic code to a Web page that calculates loan payments for a car loan.
- Create an HTML page that displays Help information.
- Use the *HyperLink* control to link one Web page to another on a Web site.
- Use the *GridView* control to display a table of database information on a Web page.
- Set the *DOCUMENT* object's *Title* property and assign a name to a Web page.

In this chapter, you'll learn how to build Web sites and Web pages by using the new Visual Web Developer tool included with Microsoft Visual Studio 2008. Visual Web Developer has the look and feel of the Visual Studio IDE, but it is customized for Web programming and Microsoft ASP.NET 3.5, the Microsoft .NET Framework component designed to provide state-of-the-art Internet functionality. ASP.NET was introduced with Microsoft Visual Studio .NET 2002 and is a replacement for WebClasses and the DHTML Page Designer in Microsoft Visual Basic 6. Although a complete description of Web programming and ASP.NET isn't possible here, there's enough in common between Web programming and Windows programming to allow you to do some useful experimentation—even if you have little or no experience with HTML. Invest a few hours in this chapter, and you'll see how quickly you can build a Web site that calculates loan payments for car loans, create an HTML page with Help information, and display loan prospects from a Microsoft Office Access database by using the *GridView* control.

Inside ASP.NET

ASP.NET 3.5 is Microsoft's latest Web development platform, and it has been enhanced in this release by improvements to the AJAX (Asynchronous JavaScript and XML) programming model, new server controls, authentication and profile services, and the *LinqDataSource* control, which allows the use of Language-Integrated Query (LINQ) in Web development contexts. Although ASP.NET has some similarities with an earlier Web programming technology named Active Server Pages (ASP), ASP.NET has been significantly enhanced since its first release in the Visual Studio .NET 2002 software, and continues to evolve as new features are added to the .NET Framework and Visual Studio software. Visual Web Developer is the tool you use to create and manage ASP.NET user interfaces, commonly called *Web pages* or (in a more comprehensive sense) *Web sites*.

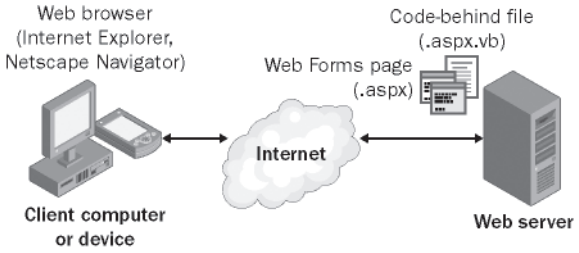
Tip In programming books, you'll sometimes see Web pages referred to as *Web forms* and Web sites referred to as *Web applications*, but those terms are less prevalent in the Visual Studio 2008 documentation.

By using Visual Web Developer, you can create a Web site that displays a user interface, processes data, and provides many of the commands and features that a standard application for Windows might offer. However, the Web site you create is viewed in a Web browser, such as Windows Internet Explorer, Mozilla Firefox, Apple Safari, or Netscape Navigator, and it is typically stored on one or more *Web servers*, which use Microsoft Internet Information Services (IIS) to display the correct Web pages and handle most of the computing tasks required by your Web site. (In Visual Studio 2005 and 2008, Web sites can also be located and run on a local computer that does not require IIS, giving you more options for development and deployment.) This distributed strategy allows your Web sites to potentially run on a wide range of Internet-based or stand-alone computers—wherever your users and their rich data sources are located.

To create a Web site in Visual Studio 2008, you click the New Web Site command on the File menu, and then use the Visual Web Developer to build one or more Web pages that will collectively represent your Web site. Each Web page consists of two pieces:

- A Web Forms page, which contains HTML and controls to create the user interface.
- A code-behind file, which is a code module that contains program code that "stands behind" the Web Forms page.

This division is conceptually much like the Windows Forms you've been creating in Visual Basic—there's a user interface component and a code module component. The code for both of these components can be stored in a single .aspx file, but typically the Web Forms page code is stored in an .aspx file, and the code-behind file is stored in an .aspx.vb file. The following illustration shows a conceptual view of how an ASP.NET Web site stored on a server is displayed in a Web browser:



In addition to Web pages, Web sites can contain code modules (.vb files), HTML pages (.htm files), configuration information (a Web.config file), global Web application information (a Global.asax file), and other components. You can use the Web Page Designer and Solution Explorer to switch back and forth between these components quickly and efficiently.

Web Pages vs. Windows Forms

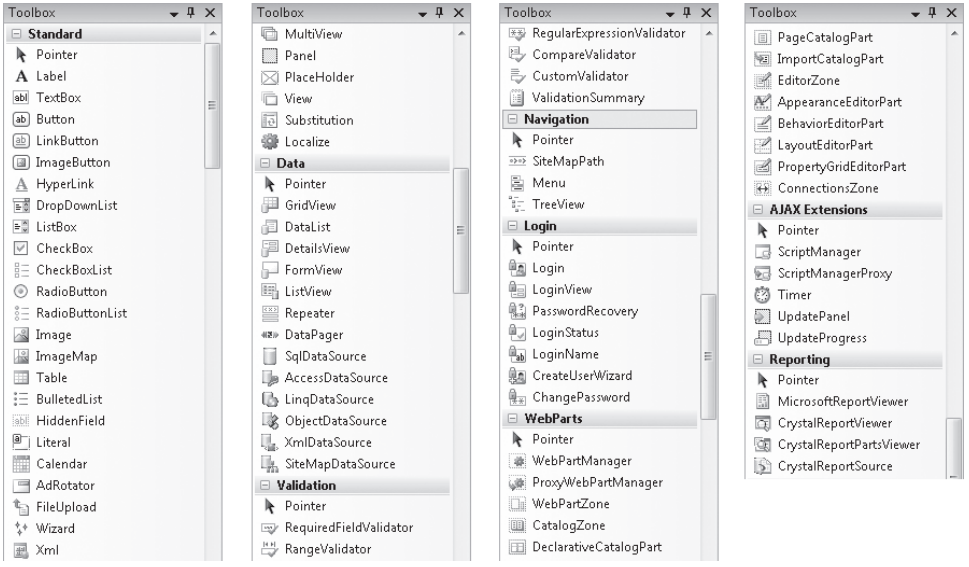
What are the important differences between Web pages and Windows Forms? To begin with, Web pages offer a slightly different programming paradigm than Windows Forms. Whereas Windows Forms use a Windows application window as the primary user interface for a program, a Web site presents information to the user through one or more Web pages with supporting program code. These pages are viewed through a Web browser, and you can create them by using the Web Page Designer.

Like a Windows Form, a Web page can include text, graphic images, buttons, list boxes, and other objects that are used to provide information, process input, or display output. However, the basic set of controls you use to create a Web page is not the set on the Common Controls tab of the Toolbox. Instead, ASP.NET Web sites must use controls on one of the tabs in the Visual Web Developer Toolbox, including Standard, Data, HTML, and many others. Each of the Visual Web Developer controls has its own unique methods, properties, and events, and although there are many similarities between these controls and Windows Forms controls, there are also several important differences. For example, Visual Studio *DataGridView* control is called *GridView* in Visual Web Developer and has different properties and methods.

Many Web page controls are *server controls*, meaning that they run on the Web server. Server controls have an "asp" prefix in their tag. HTML controls (located in the HTML tab of the Visual Web Developer Toolbox) are *client controls* by default, meaning that they run only within the end user's browser. For now, however, you simply need to know that you can use server controls, HTML controls, or a combination of both in your Web site projects. As you gain experience in Web programming, you may want to investigate AJAX programming in Visual Studio, which can enhance the efficiency of your Web applications and add advanced user-interface elements for users.

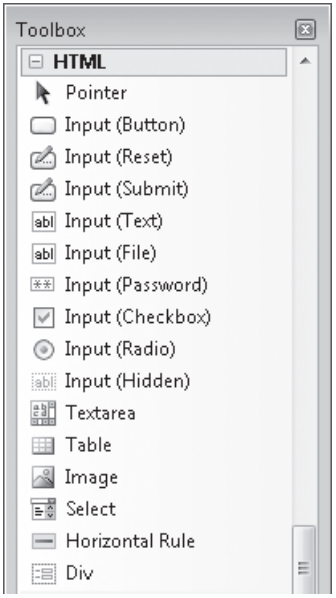
Server Controls

Server controls are more capable than HTML controls and function in many ways like the Windows Forms controls. Indeed, many of the server controls have the same names as the Windows Forms controls and offer many of the same properties, methods, and events. In addition to simple controls such as *Button*, *TextBox*, and *Label*, more sophisticated controls such as *FileUpload*, *LoginView*, and *RequiredFieldValidator* are provided on a number of tabs in the Toolbox (Visual Studio 2008 has added to the list of controls significantly). The following illustration shows most of the server controls in the Visual Web Developer Toolbox:



HTML Controls

The HTML controls are a set of older user interface controls that are supported by all Web browsers and conform closely to the early HTML standards developed for managing user interface elements on a typical Web page. They include *Button*, *Text*, and *Checkbox*—useful basic controls for managing information on a Web page that can be represented entirely with HTML code. Indeed, you might recognize these controls if you've coded in HTML before or if you've had some experience with the Visual Basic 6 DHTML Page Designer. However, although they're easy to use and have the advantage of being a "common denominator" for Web browsers, they're limited by the fact that they have no ability to maintain their own state. (In other words, the data that they contain will be lost between views of a Web page.) The following illustration shows the HTML controls offered on the HTML tab of the Toolbox in Visual Web Developer:



Building a Web Site by Using Visual Web Developer

The best way to learn about Visual Web Developer and ASP.NET is to get some hands-on practice. In the exercises in this chapter, you'll create a simple car loan calculator that determines monthly payments and displays an HTML page containing Help text. Later in the chapter, you'll use the *GridView* control to display a table of data on a Web page in the same Web site. You'll begin by verifying that Visual Studio is properly configured for ASP.NET programming, and then you'll create a new Web site project. Next you'll use the Web Page Designer to create a Web page with text and links on it, and you'll use controls in the Visual Web Developer Toolbox to add controls to the Web page.

Considering Software Requirements for ASP.NET Programming

Before you can create your first ASP.NET Web site, you need to make sure your computer is set up properly. To perform ASP.NET programming, you need to have Visual Web Developer installed. Visual Web Developer is a component of Visual Studio 2008 Standard Edition, Visual Studio 2008 Professional Edition, and Visual Studio Team System 2008 Team Suite. Visual Studio 2008 includes its own local Web server, so setting up and configuring a Web server with IIS and the .NET Framework is not required. Having a local Web server makes it easy to create and test your ASP.NET Web sites.

A useful improvement in Visual Studio 2008 is that you no longer need to develop your Web site on a computer that is fully configured to act as a Web server. In Visual Studio .NET 2002 and 2003, your development system needed to host or have access to a Web server running Windows XP Professional, Windows Server 2003, or Windows 2000 that also contained an installation of IIS, the Microsoft FrontPage 2000 Server Extensions, and the .NET Framework. This meant that if you were running Windows XP Home Edition, you were potentially out of luck, because Windows XP Home Edition does not include or support IIS. (The only workaround was to access a properly configured remote Web server.)

In Visual Studio2005 and 2008, you can create and run your Web site in one of three locations:

- Your own computer (the local file system)
- An HTTP server that contains IIS and related components
- An FTP site (a remote file server)

The first location is the option we'll use in this book, because it requires no additional hardware or software. In addition, when you develop your Web site on the local file system, all the Web site files are stored in one location. When you're finished testing the application, you can deploy the files to a Web server of your choosing.

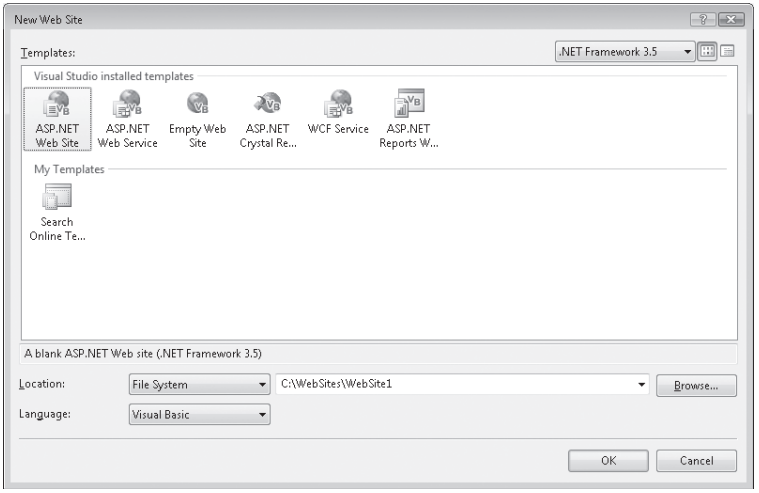
Note If you want to develop your Web site on a Web server, make sure IIS and related components are installed on your system. (On Windows XP, open Add Or Remove Programs in Control Panel, click Add/Remove Windows Components, and then browse for IIS. On Windows Vista, open Programs And Features in Control Panel, click Turn Windows Features On Or Off, and then browse for the IIS and ASP.NET features) Traditionally, Microsoft has required that you install these Web server components before the .NET Framework and Visual Studio because the .NET Framework registers extensions with IIS. If you install IIS after Visual Studio and run into problems, this could be the reason.

Create a new Web site

1. Start Visual Studio, and click the New Web Site command on the File menu.

Note If you don't see the New Web Site command on the File menu then you don't have Visual Web Developer installed.

Although you might have seen the New Web Site command before, we haven't used it yet in this book. This command starts Visual Web Developer and prepares Visual Studio to build a Web site. You see a New Web Site dialog box similar to the following:

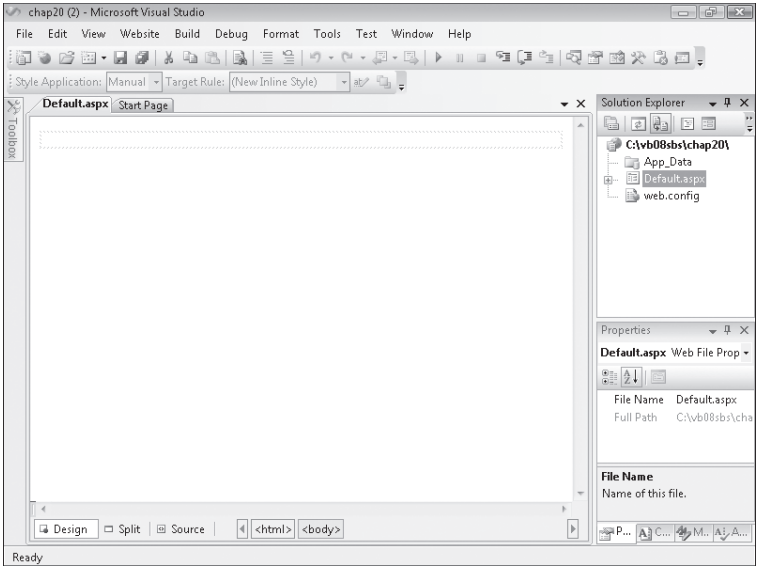


In this dialog box you can select the Web site or application template, the location for the Web site (local file system, HTTP server, or FTP site), and the programming language that you want to use (Visual Basic or Visual C#). You can also identify the version of the .NET Framework you want to target with your Web application. (Version 3.5 offers the most features, but there are times that you may need to design specifically for platforms with an earlier version of the .NET Framework.)

2. In the New Web Site dialog box, verify that ASP.NET Web Site is the selected template, and that Visual Basic is the selected language.
3. Click File System in the Location box.
4. Click Browse, create a new folder named "mychap20" in c:\vb08sbs, make sure the selected folder is set to c:\vb08sbs\mychap20 and then click Open.

   You'll notice that the Choose Location dialog box is a little different than the Project Location dialog box you've been using so far. And although you have been specifying the folder location for projects *after* you have built the projects in this book, in Visual Web Developer projects are saved up front. The "my" prefix in the pathname will avoid a conflict with the solution Web site in the practice files (c:\vb08sbs\chap20) that I've built for you.
5. Click OK in the New Web Site dialog box to finalize your changes.

   Visual Studio loads Visual Web Developer and creates a Web page (Default.aspx) to contain the user interface and a code-behind file (Default.aspx.vb) that will store the code for your Web page. (If you don't see the Web Page Designer, double-click Default.aspx in Solution Explorer now.) Your screen looks something like the one shown in the following illustration:



Unlike the Windows Forms Designer, the Web Page Designer displays the Web page in three possible views in the IDE, and three tabs at the bottom of the Designer (Design, Split, and Source) allow you to change your view of the Web page. Depending on how your system has been configured and used, you might see either the Design tab, the Split tab, or the Source tab now. (The illustration shows the Design tab.)

On the Source tab, you can view and edit the HTML code that's used to display the Web page in a Web browser. If you've used Microsoft Visual InterDev or Microsoft Office FrontPage, you'll be familiar with these two ways of displaying a Web page and perhaps with some of the HTML formatting tags that control how Web pages are actually displayed.

The Design tab shows you approximately how your Web page will look when a Web browser displays it. When the Design tab is selected, a white page appears in the Designer with the result of source-code formatting, and you can add controls to your Web page and adjust how objects on the page are arranged. The Split tab offers a composite view of the Source and Design tabs.

A few additional changes in Visual Web Developer are worth noting at this point. The Toolbox now contains several collections of controls used exclusively for Web programming. Solution Explorer also contains a different list of project files for the Web site you're building. In particular, notice the Default.aspx file in Solution Explorer; this file contains the user interface code for the active Web page. Nested under the Default.aspx file, you'll find a file named Default.aspx.vb. A configuration file named Web.config is also listed.

Now you're ready to add some text to the Web page by using the Web Page Designer.

Using the Web Page Designer

Unlike a Windows Form, a Web page can have text added directly to it when it is in the Web Page Designer. In Source view, the text appears within HTML tags somewhat like the Visual Studio Code Editor. In Design view, the text appears in top-to-bottom fashion as it does in a word processor such as Microsoft Office Word, and you'll see no HTML. In this section, you'll type text in Design view, edit it, and then make formatting changes by using buttons on the Formatting toolbar. Manipulating text in this way is usually much faster than adding a *Label* control to the Web page to contain the text. You'll practice entering the text for your car loan calculator in the following exercise.

Add text in Design view

1. Click the Design tab, if it is not currently selected, to view the Web Page Designer in Design view.

   A faint rectangle appears at the top of the Web page.

2. Position your insertion point within this rectangle.

   A blinking I-beam appears at the top of the Web page.

3. Type **Car Loan Calculator**, and then press Enter.

   Visual Studio displays the title of your Web page exactly as it will appear when you open the Web site in your browser.

4. Type the following sentence below the title:

   Enter the required information and click Calculate!

   Now you'll use the Formatting toolbar to format the title with bold formatting and a larger point size.

5. Right-click the Standard toolbar in Visual Web Developer to display the list of toolbars available in the IDE.

6. If you do not see a check mark next to Formatting in this list, click Formatting to add the Formatting toolbar.

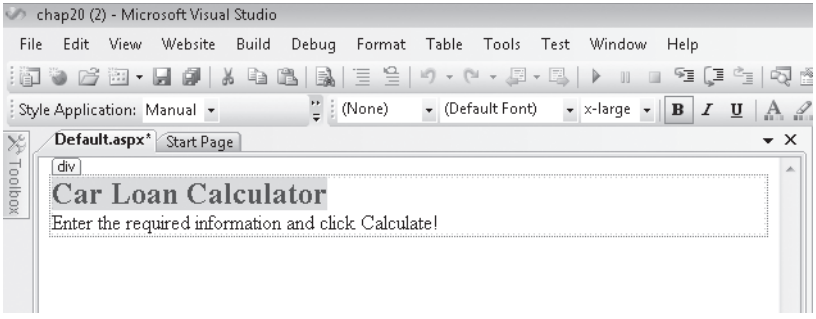
   The Formatting toolbar now appears in the IDE. Notice that it contains a few features not usually found on a text formatting toolbar.

7. Select the text "Car Loan Calculator".

   Before you can format text in Visual Web Developer, you must select it.

8. Click the Bold button on the Formatting toolbar, and set the font size to x-large 24 point.

   Your screen looks like this:

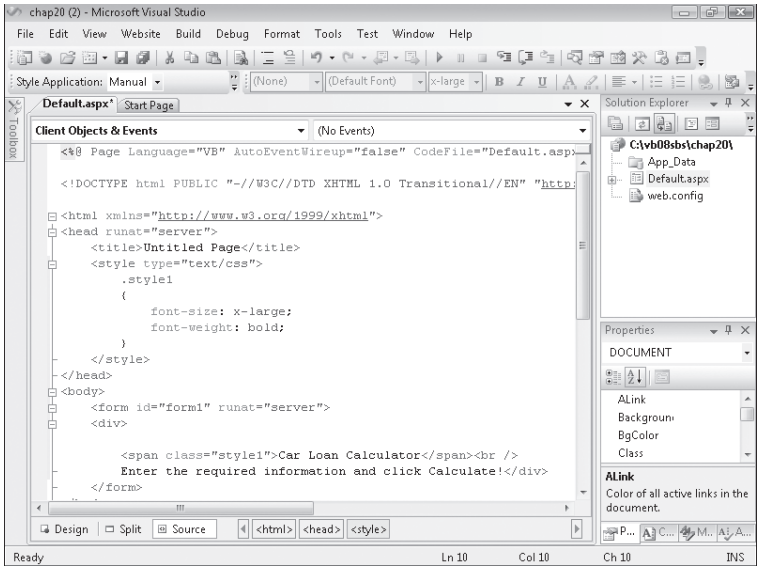


Now you'll examine the HTML code for the text and formatting you entered.

View the HTML for a Web page

1. Click the Source tab at the bottom of the Designer.

   The Source tab displays the actual HTML code for your Web page. To see more of the code, you might want to temporarily resize a few programming tools. The HTML code looks like the following illustration. Your HTML code might have some differences.



   A Web page is made up of file and document information, formatting codes called HTML tags that are enclosed in angle brackets, and the text and objects to be displayed by your Web page. This Web page is still rather short—it contains a header with information about the language you selected when creating the Web application, the name of any code-behind file, and any inherited forms.

   HTML tags typically appear in pairs so that you can see clearly where a section begins and ends. For example, the *<body>* tag identifies the beginning of the document and the *</body>* tag identifies the end. Notice that the "Car Loan Calculator" text appears below a block of HTML style code that formats the text as bold with a font size of x-large (24 points). Below the "Car Loan Calculator" text the second line of text you entered is displayed.

> **Tip** Remember that the Source tab is an actual editor, so you can change the text you entered by using standard text editing techniques. If you know something about HTML, you can add additional formatting tags and content as well.

2. Click the Design tab to display your Web page in Design view, and open the Toolbox if it is not visible.

Adding Server Controls to a Web Site

Now you'll add *TextBox*, *Label*, and *Button* controls to the car loan calculator. Although these controls are located in the Visual Web Developer Toolbox, they're very similar to the Windows Forms controls of the same name that you've used throughout this book. (I'll cover a few of the important differences as they come up.) The most important thing to remember is that in the Web Page Designer, controls are inserted at the insertion point if you double-click the control name in the Toolbox. After you add the controls to the Web page, you'll set property settings for the controls.

Use *TextBox*, *Label*, and *Button* controls

1. Display the Standard tab of the Toolbox, if it isn't already visible.
2. Position the insertion point to the end of the second line of text on the Web page, and then press the Enter key three times to create a little blank space below the text for the controls.

   Because controls are placed at the insertion point, you need to use the text editing keys to position the insertion point appropriately before double-clicking a control in the Toolbox.

> **Note** By default, the Web Page Designer positions controls relative to other controls. This is an important difference between the Web Page Designer and the Windows Forms Designer. The Windows Forms Designer allows you to position controls wherever you like on a form. You can change the Web Page Designer so that you can position controls wherever you like on a Web page (called absolute positioning); however, you might get different behavior in different Web browsers.

3. Double-click the *TextBox* control on the Standard tab of the Toolbox to create a text box object at the insertion point on the Web page.

   Notice the *asp:textbox#TextBox1* text that appears above the text box object. The "asp" prefix indicates that this object is an ASP.NET server control. (This text disappears when you run the program.)
4. Click the right side of the text box object to place the insertion point at the outside edge, and then press Enter twice.
5. Double-click the *TextBox* control again to add a second text box object to the Web page.

6. Repeat steps 4 and 5 to create a third text box object below the second text box.

   Now you'll use the *Label* control to insert labels that identify the purpose of the text boxes.

7. Click to the right of the first text box object to place the insertion point at the right edge of the text box.

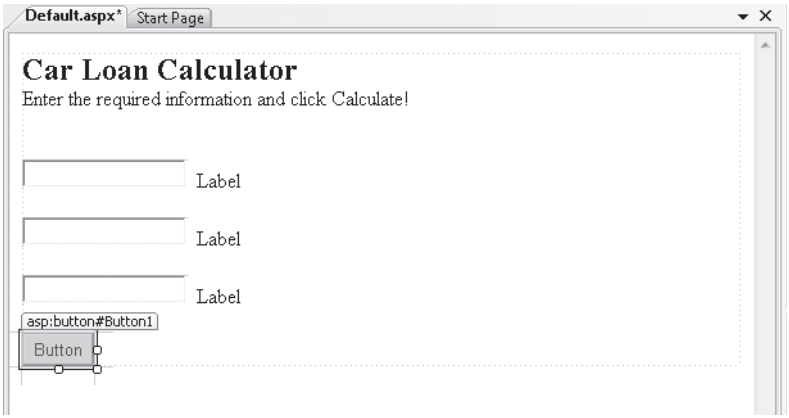
8. Press Spacebar twice to add two blank spaces, and then double-click the *Label* control in the Toolbox to add a label object to the Web page.

9. Repeat steps 7 and 8 to add label objects to the right of the second and third text boxes.

10. Click to the right of the third label object to place the insertion point to the right of the label, and then press Enter twice.

11. Double-click the *Button* control to create a button object at the bottom of the Web page.

    The *Button* control, like the *TextBox* and *Label* controls, is very similar to its Windows Forms counterpart. Your screen looks like this:

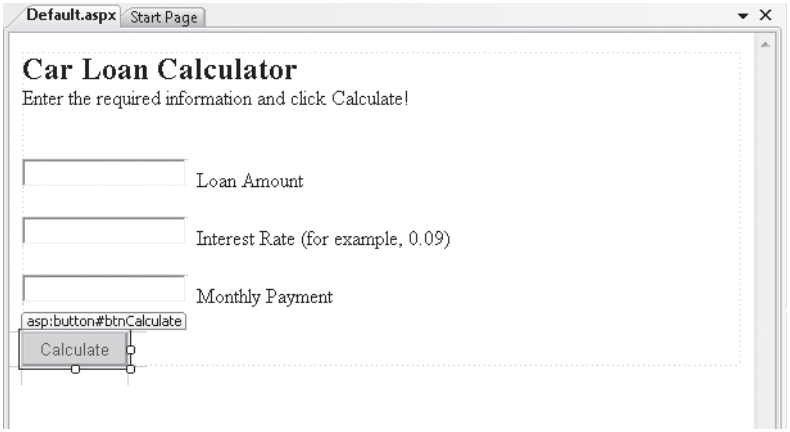


Now you'll set a few properties for the seven new controls you have created on the Web page. Open the Properties window if it is not visible (press F4). As you set the properties, you'll notice one important difference between Web pages and Windows Forms—the familiar Name property has been changed to ID in Visual Web Developer. Despite their different names, the two properties perform the same function.

12. Set the following properties for the objects on the form:

Object	Property	Setting
TextBox1	*ID*	txtAmount
TextBox2	*ID*	txtInterest
TextBox3	*ID*	txtPayment
Label1	*ID*	lblAmount
	Text	"Loan Amount"
Label2	*ID*	lblInterest
	Text	"Interest Rate (for example, 0.09)"
Label3	*ID*	lblPayment
	Text	"Monthly Payment"
Button1	*ID*	btnCalculate
	Text	"Calculate"

Your Web page looks like this:



Writing Event Procedures for Web Page Controls

You write default event procedures (or event handlers) for controls on a Web page by double-clicking the objects on the Web page and typing the necessary program code in the Code Editor. Although the user will see the controls on the Web page in his or her own Web browser, the actual code that's executed will be located on the local test machine or a Web server, depending on how you configured your project for development and how it is eventually deployed. For example, when the user clicks a button on a Web page that is hosted by a Web server, the browser sends the button click event back to the server, which processes the event and sends a new Web page back to the browser. Although the process seems similar to that of Windows Forms, there's actually a lot going on behind the scenes when a control is used on an ASP.NET Web page!

In the following exercise, you'll practice creating the default event procedure for the *btnCalculate* object on the Web page.

Create the *btnCalculate_Click* event procedure

1. Double-click the Calculate button on the Web page.

   The code-behind file (Default.aspx.vb) opens in the Code Editor, and the *btnCalculate_Click* event procedure appears.

2. Type the following program code:

```
Dim LoanPayment As Double
'Use Pmt function to determine payment for 36 month loan
LoanPayment = Pmt(CDbl(txtInterest.Text) / 12, 36, CDbl(txtAmount.Text))
txtPayment.Text = Format(Abs(LoanPayment), "$0.00")
```

   This event procedure uses the *Pmt* function, a financial function that's part of the Visual Basic language, to determine what the monthly payment for a car loan would be by using the specified interest rate (*txtInterest.Text*), a three-year (36-month) loan period, and the specified principal amount (*txtAmount.Text*). The result is stored in the *LoanPayment* double-precision variable, and then it is formatted with appropriate monetary formatting and displayed by using the *txtPayment* text box object on the Web page.

The two *Text* properties are converted from string format to double-precision format by using the *CDbl* function. The *Abs* (absolute value) function is used to make the loan payment a positive number. (*Abs* currently has a jagged underline in the Code Editor because it relies on the *System.Math* class, which you'll specify next.) Why make the loan payment appear as a positive number? The *Pmt* function returns a negative number by default (reflecting money that's owed), but I think negative formatting looks strange when it isn't part of a balance sheet, so I'm converting it to positive.

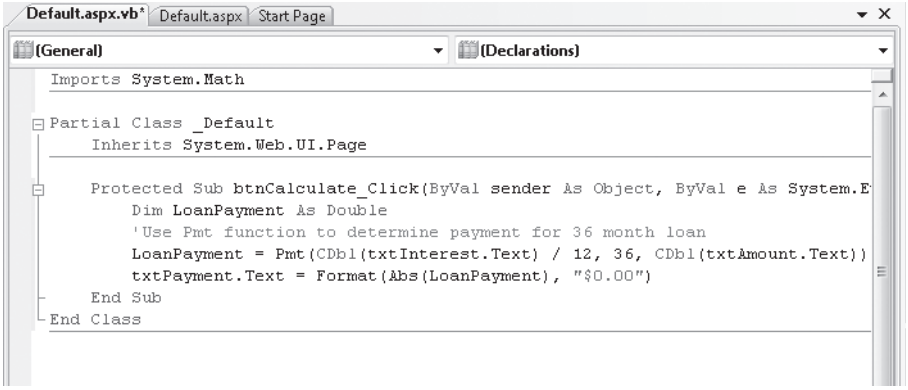
Notice that the program statements in the code-behind file are just regular Visual Basic code—the same stuff you've been using throughout this book. Basically, the process feels similar to creating a Windows application.

3. Scroll to the top of the Code Editor, and enter the following program statement as the first line of the file:

```
Imports System.Math
```

As you learned in Chapter 5, "Visual Basic Variables and Formulas, and the .NET Framework," the *Abs* function isn't included in Visual Basic by default, but it's part of the *System.Math* class in the .NET Framework and can be more easily referenced in your project by the *Imports* statement. Web applications can make use of the .NET Framework class libraries just as Windows applications can.

The Code Editor looks like this:



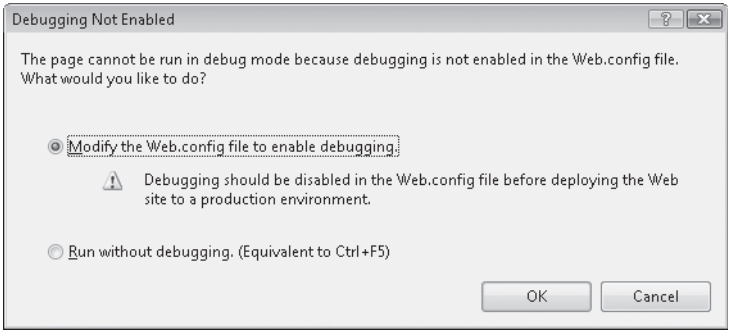
4. Click the Save All button on the Standard toolbar.

That's it! You've entered the program code necessary to run the car loan calculator and make your Web page interactive. Now you'll build and run the project and see how it works. You'll also learn a little bit about security settings within Internet Explorer, a topic closely related to Web development.

Build and view the Web site

1. Click the Start Debugging button on the Standard toolbar.

   Visual Studio displays the following message about debugging:



   This potentially confusing dialog box is not a major concern. It just indicates that the Web.config file in your project does not currently allow debugging (a standard security feature). Although you can bypass this dialog box each time you test the application within Visual Studio by clicking the Run Without Debugging button, I recommend that you modify the Web.config file now.

Security Tip Before you widely distribute or deploy a real Web site, be sure to disable debugging in Web.config to keep your application safe from unauthorized tampering.

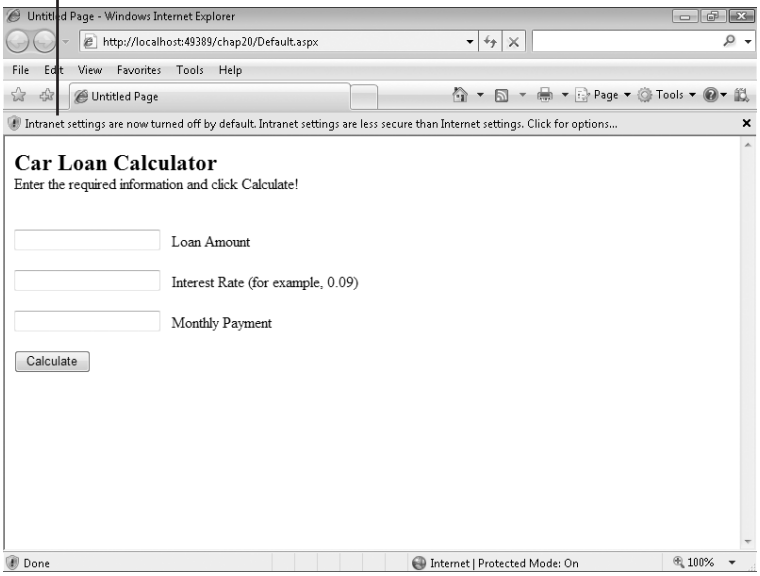
2. Click OK to modify the Web.config file.

   Visual Studio modifies the file, builds your Web site, and displays the opening Web page in Internet Explorer.

Security Tip If Internet Explorer displays the message "Script Debugging Disabled," click Yes to continue. You can adjust a security setting within Internet Explorer so that this message does not appear in the future. (We won't be debugging right now.) You can modify the Internet Explorer Disable Script Debugging setting by clicking the Internet Options command on the Tools menu, clicking the Advanced tab, and clicking to clear the Disable Script Debugging option.

The car loan calculator looks like the following illustration. If Internet Explorer does not appear, you might need to select it on the Windows taskbar.

When you first run your Web application in Internet Explorer, you may see a security warning.



Security Tip You might see the Information Bar at the top of Internet Explorer indicating that intranet settings are turned off by default. (This Information Bar is shown in the previous illustration.) An intranet warning is again related to Internet Explorer's desire to protect you from rogue programs or unauthorized access. An intranet is a local network (typically a home network or small workgroup network), and because Visual Studio uses intranet-style addressing when you test Web sites built on your own computer, you're likely to see the warning message. To temporarily suppress the warning, click the Information Bar and then click Don't Show Me This Again. To remove intranet warnings more permanently, click the Internet Options command on the Internet Explorer Tools menu, click the Security tab, and then click Local Intranet. Click the Sites button, and clear the check mark from Automatically Detect Intranet Network in the Local Intranet dialog box. However, exercise caution whenever you disable security warnings, as they are meant to protect you.

Tip When you started this Web site, you might have noticed a balloon pop up in the notification area of the Windows taskbar. This balloon indicates that the local Web server has started to run this Web site. If you right-click the ASP.NET Development Server icon in the notification area, you can get more information about the Web server.

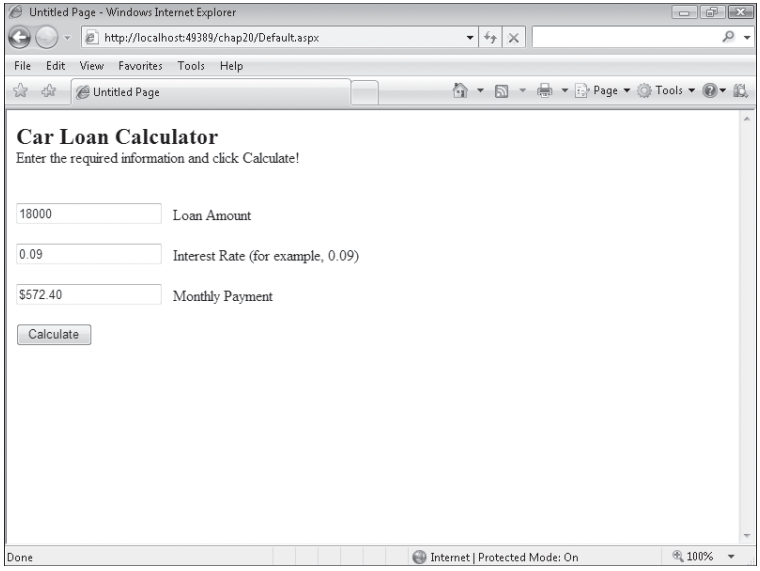
Now let's get back to testing our Web page.

3. Type **18000** in the Loan Amount text box, and then type **0.09** in the Interest Rate text box.

   You'll compute the monthly loan payment for an $18,000 loan at 9 percent interest for 36 months.

4. Click the Calculate button.

   Visual Basic calculates the payment amount and displays $572.40 in the Monthly Payment text box. Your screen looks like this:



5. Close Internet Explorer.

   You're finished testing your Web site for now. When Internet Explorer closes, your program is effectively ended. As you can see, building and viewing a Web site is basically the same as building and running a Windows application, except that the Web site is executed in the browser. You can even set break points and debug your application just as you can in a Windows application.

Curious about installing a Web site like this on an actual Web server? The basic procedure for deploying Web sites is to copy the .aspx files and any necessary support files for the project to a properly configured virtual directory on a Web server running IIS and the .NET Framework. There are a couple of ways to perform deployment in Visual Web Developer. To get started, click Copy Web Site on the Web site menu, or click Publish Web Site on the Build menu. For more information, see "ASP.NET Deployment Overview" in the Visual Studio documentation.

Validating Input Fields on a Web Page

Although this Web page is useful, it runs into problems if the user forgets to enter a principal amount or an interest rate or specifies data in the wrong format. To make Web sites like this more robust, I usually add one or more *validator controls* that force users to enter input in the proper format. The validator controls are located on the Validation tab of the Visual Web Developer Toolbox and include controls that require data entry in a field (*RequiredFieldValidator*), require entry in the proper range (*RangeValidator*), and so on. For information on the validator controls, search the Visual Studio documentation. They are straight forward to use.

Adding Additional Web Pages and Resources to a Web Site

Now the fun begins! Only very simple Web sites consist of just one Web page. Using Visual Web Developer, you can quickly expand your Web site to include additional information and resources, including HTML pages, XML pages, text files, database records, Web services, site maps, and more. If you want to add an HTML page (a standard Web page containing text and HTML client-side controls), you have two options.

- You can create a new HTML page by using the Add New Item command on the Website menu. After you create the HTML page, you add text and HTML objects to the page by using the Web Page Designer.
- You can add an HTML page that you have already created by using the Add Existing Item command on the Web site menu, and then customize the page in the Web Page Designer. You use this method if you want to include one or more Web pages that you have already created in a tool such as Microsoft Expression Web. (If possible, add pages that don't rely on external style sheets and resources, or you'll need to add those items to the project as well.)

To link pages together, Visual Web Developer provides the *HyperLink* control, which creates a hyperlink label object that the user clicks to jump from the current Web page to a new one. When you use a *HyperLink* control, you set the text that will be displayed on the page by using the *Text* property, and you specify the desired resource to jump to (either a URL or a local path) by using the *NavigateUrl* property.

In the following exercise, you'll create a second Web page by using the Add New Item command, and you'll save it in HTML format along with your other project files. The new page will be a Help file that users of your Web site can access to get operating instructions for the loan calculator. After you create the new page, you'll add a *HyperLink* control to the first page and set the *HyperLink* control's *NavigateUrl* property to the new HTML page.

Create an HTML page

1. Click the Add New Item command on the Web site menu.

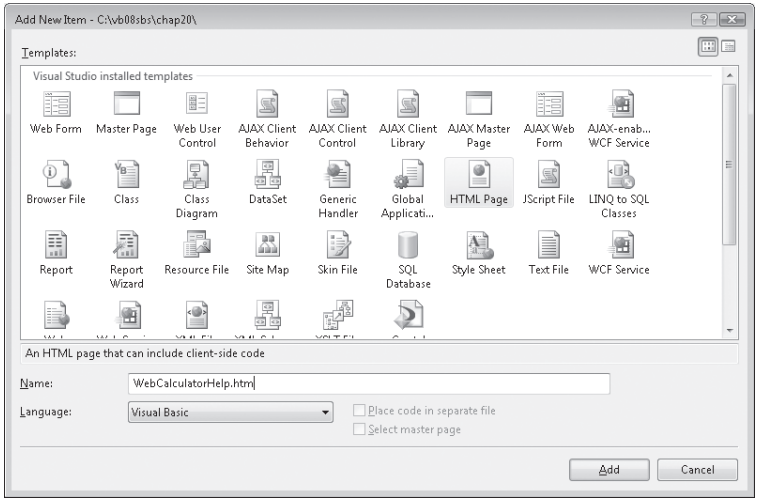
   The Add New Item dialog box opens, allowing you to add a number of different Internet resources to your Web site.

2. Click the HTML Page template.

   You'll insert a blank HTML page into the project, which you can use to display formatted text and HTML controls. (You cannot add server controls to this page, because simple HTML pages are controlled by the client's browser, not a Web server.)

3. Type **WebCalculatorHelp.htm** in the Name text box.

Your screen looks like this:



4. Click Add.

   The WebCalculatorHelp.htm file is added to Solution Explorer and is opened in the Web Page Designer in Design view.

   Notice that only HTML controls are displayed in the Toolbox. Because this is an HTML page, the server controls aren't supported.

5. If necessary, click the Design tab to display the HTML page in Design view.

   The I-beam insertion point blinks on the page, ready for your input.

6. Type the following text:

   Car Loan Calculator

   The Car Loan Calculator Web site was developed for the book *Microsoft Visual Basic 2008 Step by Step*, by Michael Halvorson (Microsoft Press, 2008). The Web site is best viewed using Microsoft Internet Explorer version 6.0 or later. To learn more about how this ADO.NET application was created, read Chapter 20 in the book.

   Operating Instructions:

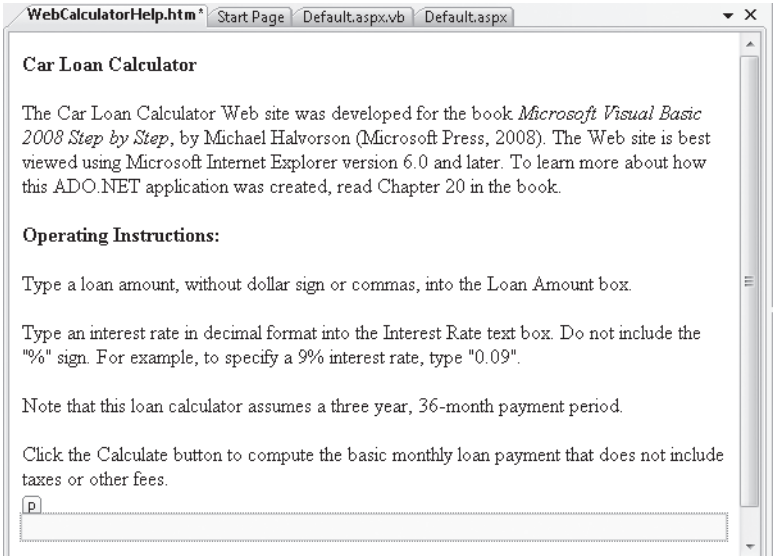
   Type a loan amount, without dollar sign or commas, into the Loan Amount box.

   Type an interest rate in decimal format into the Interest Rate text box. Do not include the "%" sign. For example, to specify a 9% interest rate, type "0.09".

   Note that this loan calculator assumes a three-year, 36-month payment period.

   Click the Calculate button to compute the basic monthly loan payment that does not include taxes or other fees.

7. Using buttons on the Formatting toolbar, add bold and italic formatting, as shown here:



8. Click the Save All button on the Standard toolbar to save your changes.

Now you'll use the *HyperLink* control to create a hyperlink on the first Web page that opens the WebCalculatorHelp.htm file.

Use the *HyperLink* control

1. Display the Car Loan Calculator Web page (Default.aspx) in Design view.
2. Place the insertion point to the right of the button object on the Web page, and then press Enter twice.
3. Double-click the *HyperLink* control on the Standard tab of the Toolbox to create a hyperlink object at the insertion point.
4. Set the *Text* property of the hyperlink object to "Get Help".

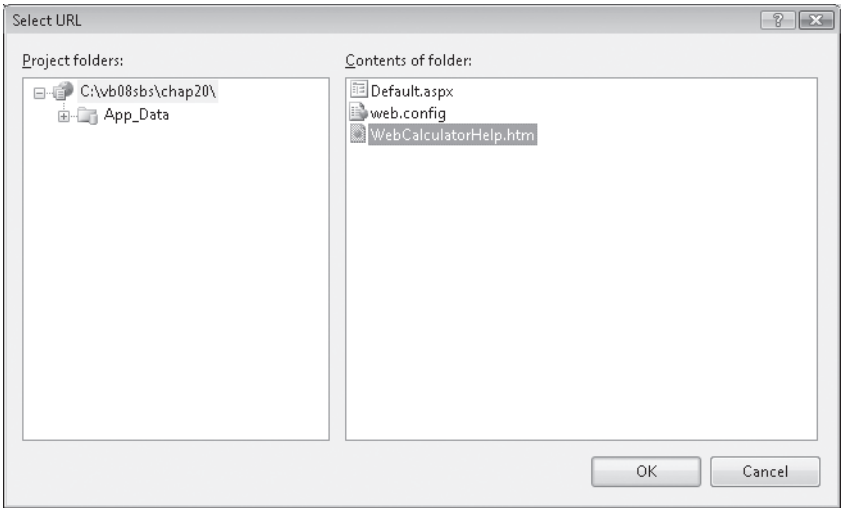
   The *Text* property contains the text that will appear as the underlined hyperlink on the Web page. You want to use words here that will make it obvious that there's a Web page available containing Help text.
5. Set the *ID* property of the hyperlink object to "lnkHelp".

   Naming this object makes it consistent with the other objects in the Web site.
6. Click the *NavigateUrl* property, and then click the ellipsis (...) button in the second column.

   Visual Studio opens the Select URL dialog box, which prompts you for the location of the Web page to which you want to link.

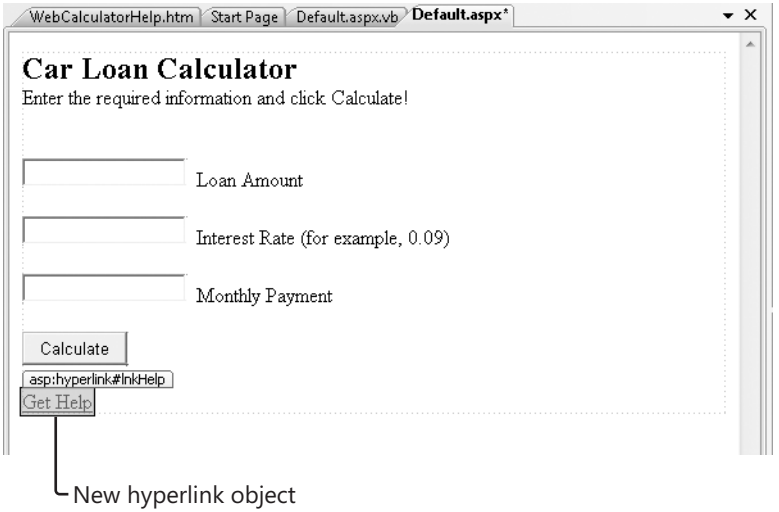
7. Click the WebCalculatorHelp.htm file in the Contents Of Folder list box.

   The URL text box displays the name of the file you want to use as the hyperlink. Your dialog box looks like this:



8. Click OK to set the *NavigateUrl* property.

   Your Web page looks like this:



   Your link is finished, and you're ready to view the Web site in your browser again.

9. Click the Save All button.

10. Click the Start Debugging button.

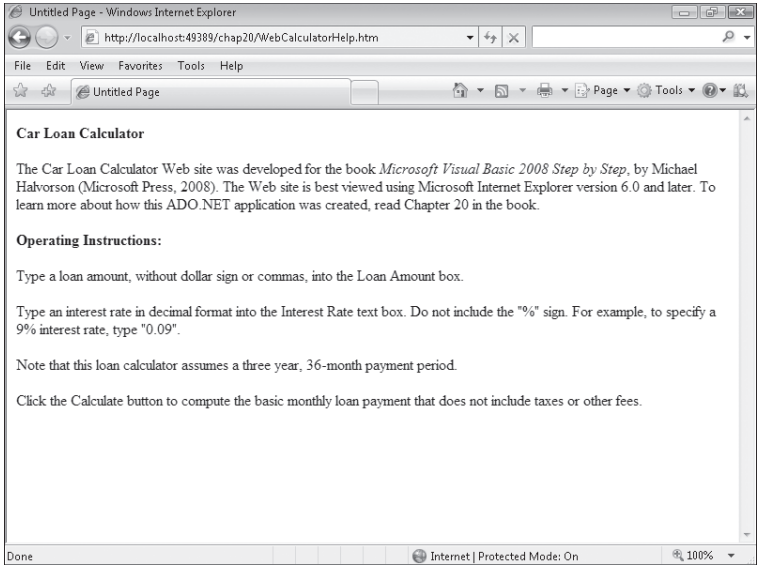
    Visual Studio builds the Web site and displays it in Internet Explorer.

11. Compute another loan payment to experiment further with the loan calculator.

    If you want to test another set of numbers, try entering 20000 for the loan amount and 0.075 for the interest rate. The result should be $622.12.

12. Now click the Get Help hyperlink to see how the *HyperLink* control works.

    Internet Explorer displays your new HTML page on the screen. Your HTML page looks something like this:



13. Read the text, and then click the Back button in Internet Explorer.

    Just like any Web site, this one lets you click the Back and Forward buttons to jump from one Web page to the next.

14. Close Internet Explorer to close the Web site.

    You've added a simple HTML page to your Web site, and you have experimented with using the *HyperLink* control to link together Web pages. Pretty cool. Now try something more sophisticated that shows how far you can take your Web site if you choose to include information from a database.

Displaying Database Records on a Web Page

For many users, one of the most exciting aspects of the World Wide Web is the ability to access large amounts of information rapidly through a Web browser. Often, of course, the quantity of information that needs to be displayed on a commercial Web site far exceeds what a developer can realistically prepare using simple text documents. In these cases, Web programmers add database objects to their Web sites to display tables, fields, and records of database information on Web pages, and they connect the objects to a secure database residing on the Web server or another location.

Visual Studio 2008 makes it easy to display simple database tables on a Web site, so as your computing needs grow, you can use Visual Studio to process orders, handle security, manage complex customer information profiles, and create new database records—all from the Web. Importantly, Visual Web Developer delivers this power very effectively. For example, by using the *GridView* control, you can display a database table containing dozens or thousands of records on a Web page without any program code. You'll see how this works by completing the following exercise, which adds a Web page containing loan contact data to the Car Loan Calculator project. If you completed the database programming exercises in Chapter 18, "Getting Started with ADO.NET," and Chapter 19, "Data Presentation Using the *DataGridView* Control," be sure to notice the similarities (and a few differences) between database programming in a Windows environment and database programming on the Web.

Add a new Web page for database information

1. Click the Add New Item command on the Web site menu.

   Visual Web Developer displays a list of components that you can add to your Web site.

2. Click the Web Form template, type **InstructorLoans.aspx** in the Name text box, and then click Add.

   Visual Web Developer adds a new Web page to your Web site. Unlike the HTML page you added earlier, this Web page component is capable of displaying server controls.

3. If necessary, click the Design tab to switch to Design view.

4. Enter the following text at the top of the Web page:

   The following grid shows instructors who want loans and their contact phone numbers:

5. Press Enter twice to add two blank lines below the text.

   Remember that Web page controls are added to Web pages at the insertion point, so it is always important to create a few blank lines when you are preparing to add a control.

Next you'll display two fields from the *Instructors* table of the Students.mdb database by adding a *GridView* control to the Web page. *GridView* is similar to the *DataGridView* control you used in Chapter 19, but *GridView* has been optimized for use on the Web. (There are also a few other differences, which you can explore by using the Properties window and Visual Studio documentation.) Note that I'm using the same Access database table I used in Chapters 18 and 19, so you can see how similar database programming is in Visual Web Developer. Many programmers also use SQL databases on their Web sites, and Visual Web Developer also handles that format very well.

Add a *GridView* control

1. With the new Web page open and the insertion point in the desired location, double-click the *GridView* control on the Data tab of the Visual Web Developer Toolbox.

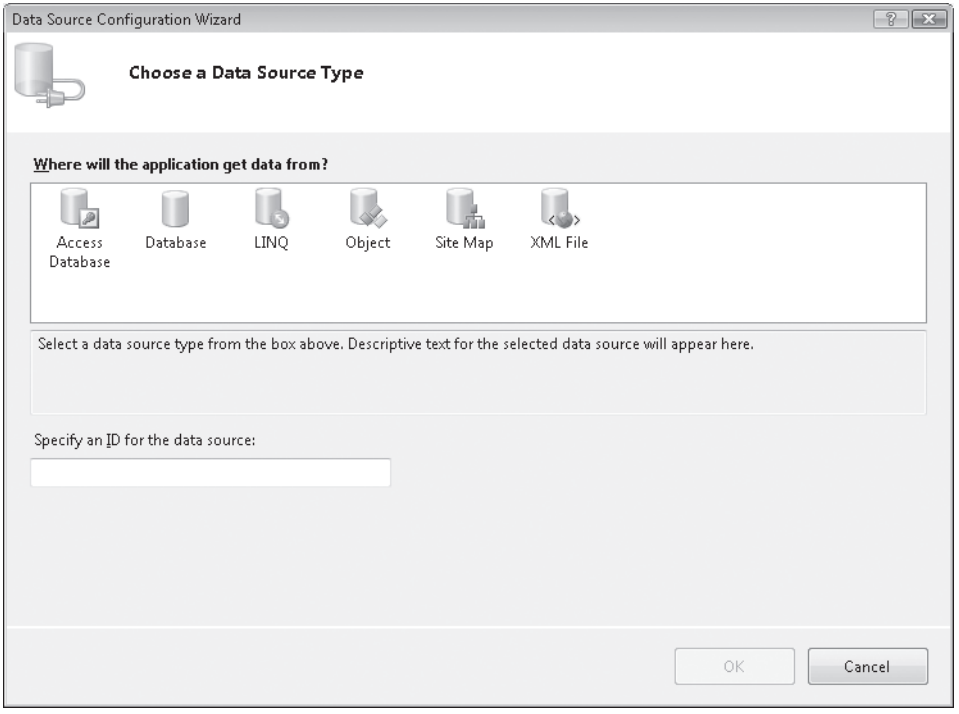
   Visual Web Developer adds a grid view object named *GridView1* to the Web page. The grid view object currently contains placeholder information.

2. If the Common GridView Tasks list is not already displayed, click the *GridView1* object's shortcut arrow to display the list.

   Click the Choose Data Source arrow, and then click the <New Data Source> option.

3. Visual Web Developer displays the Data Source Configuration Wizard, a tool that you used in Chapters 18 and 19 to establish a connection to a database and select the tables and fields that will make up a dataset.

Your screen looks like this:



4. Click the Access Database icon, type **Students** in the Specify An ID For The Data Source box, and then click OK.

   You are now prompted to specify the location of the Access database on your system. (This dialog box is slightly different than the one you used in Chapter 18.)

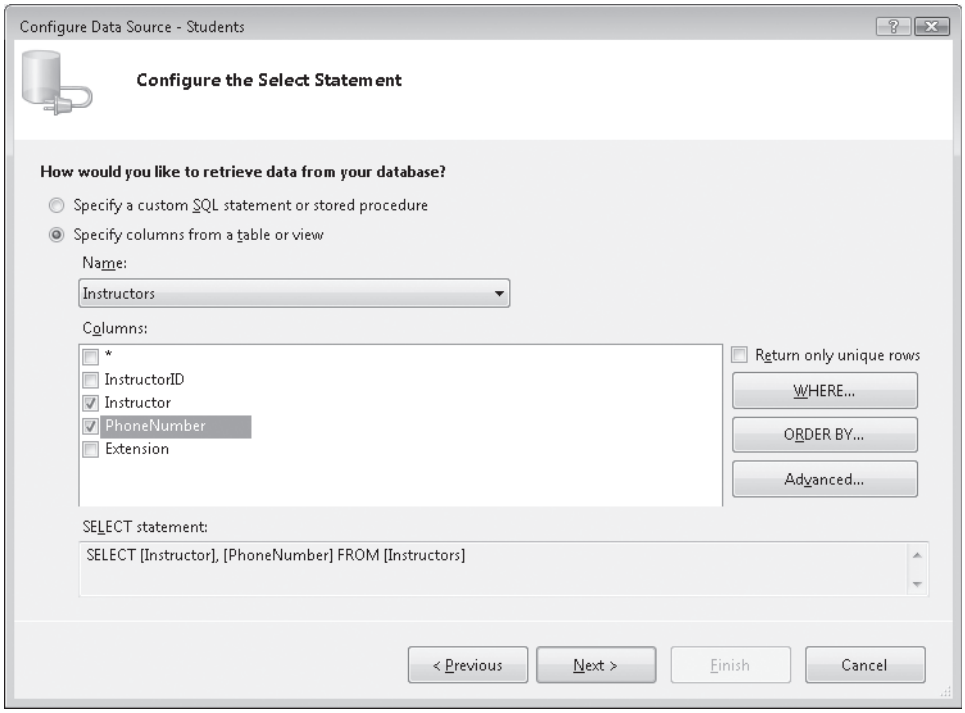
5. Type c:\vb08sbs\chap18\students.mdb, and then click Next.

   You are now asked to configure your data source; that is, to select the table and fields that you want to display on your Web page. Here you'll use two fields from the *Instructors* table. (Remember that in Visual Studio, database fields are often referred to as columns, so you'll see the word "columns" used in the IDE and the instructions below.)

6. Click the Name arrow, and then click Instructors in the list box.

7. Select the Instructor and PhoneNumber check boxes in the Columns list box.

   Your screen looks like this:



   Through your actions here, you are creating an SQL SELECT statement that configures a dataset representing a portion of the Students.mdb database. You can see the SELECT statement at the bottom of this dialog box.

8. Click Next to see the Test Query screen.

9. Click the Test Query button to see a preview of your data.

   You'll see a preview of actual Instructor and PhoneNumber fields from the database. This data looks as expected, although if we were preparing this Web site for wider distribution, we would take the extra step of formatting the PhoneNumber column so that it contains standard spacing and phone number formatting.

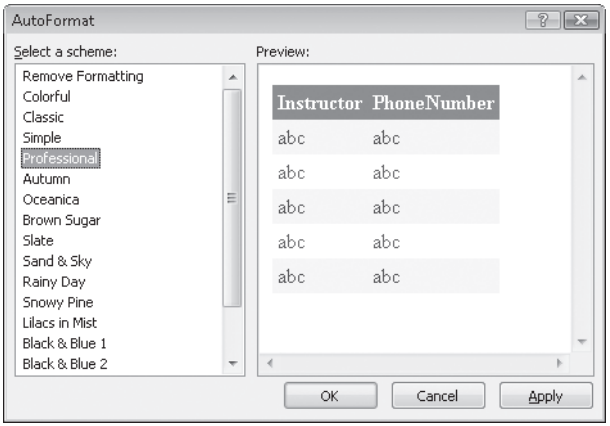
10. Click Finish.

Visual Web Developer closes the wizard and adjusts the number of columns and column headers in the grid view object to match the selections that you have made. However, it continues to display placeholder information ("abc") in the grid cells.

11. With the Common GridView Tasks list still open, click the Auto Format command.

12. Click the Professional scheme.

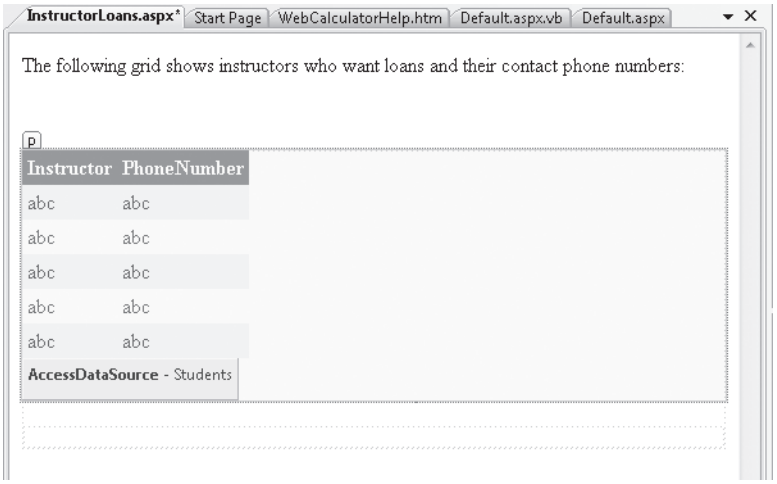
The Auto Format dialog box looks like this:



The ability to quickly format, adjust, and preview formatting options is a great feature of the *GridView* control.

13. Click OK and then close the Common GridView Tasks list.

The InstructorLoans.aspx Web page is complete now, and looks like the following illustration. (My *GridView* control is within a <*p*> tag, but yours might be within a <*div*> tag.)



Now you'll add a hyperlink on the first Web page (or home page) that will display this Web page when the user wants to see the database table. You'll create the hyperlink with the *HyperLink* control.

Add a hyperlink to the home page

1. Click the Default.aspx tab at the top of the Designer.

   The home page for your Web site opens in the Designer.

2. Click to the right of the Get Help (*lnkHelp*) object to place the insertion point after that object.

3. Press Enter twice to create space for a second hyperlink.

4. Double-click the *HyperLink* control on the Standard tab of the Toolbox to create a hyperlink object at the insertion point.

5. Set the *Text* property of the hyperlink object to "Display Loan Prospects".

   We'll pretend that your users are bank loan officers (or well-informed car salespeople) looking to sell auto loans to university professors. Display Loan Prospects will be the link they click to view the selected database records.

6. Set the *ID* property of the hyperlink object to "lnkProspects".

7. Click the *NavigateUrl* property, and then click the ellipsis button.

   Visual Studio opens the Select URL dialog box.

8. Click the InstructorLoans.aspx file in the Contents Of Folder list box, and then click OK.

Your link is finished, and you're ready to test the Web site and *GridView* control in your browser.

Test the final Car Loan Calculator Web site

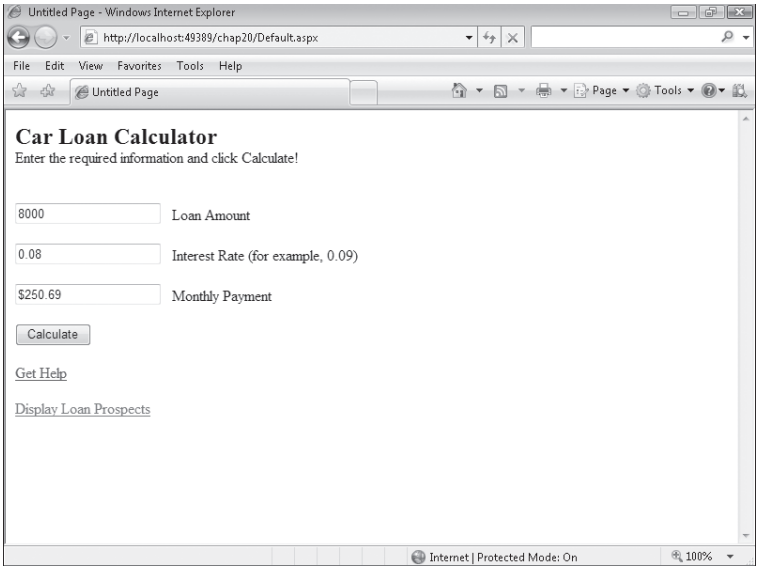
Tip The complete Car Loan Calculator Web site is located in the c:\vb08sbs\chap20\chap20 folder. Use the Open Web Site command on the File menu to open an existing Web site.

1. Click the Start Debugging button.

   Visual Studio builds the Web site and displays it in Internet Explorer.

2. Enter **8000** for the loan amount and **0.08** for the interest rate, and then click Calculate.

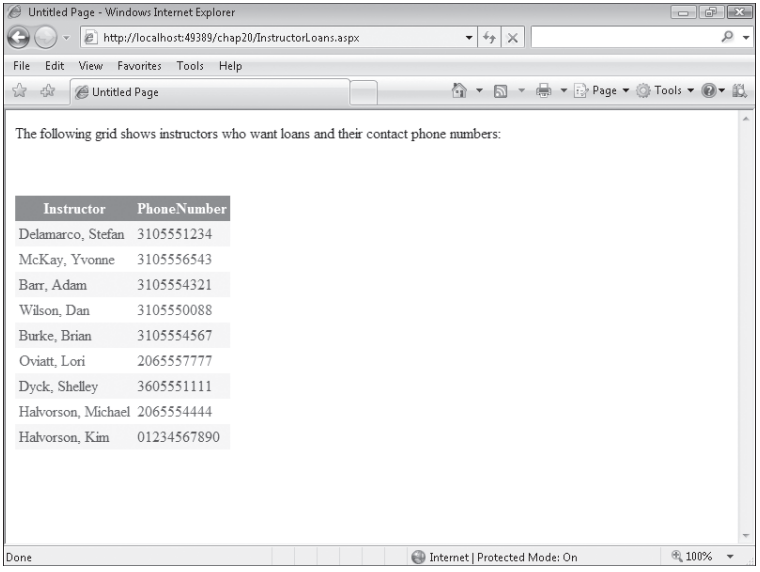
   The result is $250.69. Whenever you add to a project, it is always good to go back and test the original features to verify that they have not been modified inadvertently. Your screen looks like the illustration shown on the following page.



The new hyperlink (Display Loan Prospects) is visible at the bottom of the Web page.

3. Click Display Loan Prospects to load the database table.

   Internet Explorer loads the *Instructor* and *PhoneNumber* fields from the Students.mdb database into the grid view object. Your Web page looks something like this:



The information is nicely formatted and appears useful. By default, you'll find that the data in this table cannot be sorted, but you can change this option by selecting the Enable Sorting check box in Common GridView Tasks. If your database contains many rows (records) of information, you can select the Enable Paging check box in Common GridView Tasks to display a list of page numbers at the bottom of the Web page (like a list you might see in Microsoft Document Explorer or a search engine that displays many pages of "hits" for your search).

4. Click the Back and Forward buttons in Internet Explorer.

   As you learned earlier, you can jump back and forth between Web pages in your Web site, just as you would in any professional Web site.

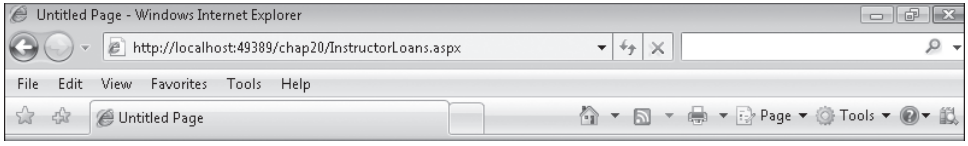
5. When you're finished experimenting, close Internet Explorer to close the Web site.

You've added a table of custom database information without adding any program code!

One Step Further: Setting the Web Site Title in Internet Explorer

Haven't had enough yet? Here's one last Web programming tip to enhance your Web site and send you off on your own explorations.

You might have noticed while testing the Car Loan Calculator Web site that Internet Explorer displayed "Untitled Page" in the title bar and window tab when displaying your Web site. In other words, your screen looked like this:



You can customize what Internet Explorer and other browsers display in the title bar by setting the *Title* property of the *DOCUMENT* object for your Web page. Give it a try now.

Set the *Title* property

1. With the Default.aspx Web page open in Design view, click the *DOCUMENT* object in the Object list box at the top of the Properties window.

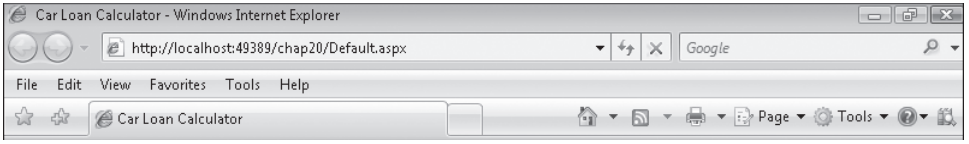
Each Web page in a Web site contains a *DOCUMENT* object that holds important general settings for the Web page. However, the *DOCUMENT* object is not selected by default in the Designer, so you might not have noticed it. One of the important properties for the *DOCUMENT* object is *Title*, which sets the title of the current Web page in the browser.

2. Set the *Title* property to "Car Loan Calculator".

   The change does not appear on the screen, but Visual Web Developer records it internally.

3. Click the Start Debugging button.

   Visual Studio opens Internet Explorer and loads the Web site. Now a more useful title bar appears, as shown in the following illustration:



   Now that looks better.

4. Close Internet Explorer, and update the *Title* properties for the other Web pages in your Web site.

5. When you're finished experimenting with the Car Loan Calculator, save your changes, and close Visual Studio.

Congratulations on completing the entire *Microsoft Visual Basic 2008 Step by Step* programming course! Take a few moments to flip back through this book and see all that you have learned. Now you're ready for more sophisticated Visual Basic challenges and programming techniques. Check out the resource list in the Appendix, "Where to Go For More Information," for a few ideas about continuing your learning. But take a break first—you've earned it!

Chapter 20 Quick Reference

To	Do this
Create a new ASP.NET Web site	Click the New Web Site command on the File menu, click the ASP.NET Web Site template, specify a folder location in the Location list box, and then click OK.
Switch between Design view and Source view in the Web Page Designer	Click the Source or Design tabs in the Web Page Designer. For a mixed view, click the Split tab.
Enter text on a Web page	Click the Design tab, and then type the text you want to add.
Format text on a Web page	On the page, select the text that you want to format, and then click a button or control on the Formatting toolbar.
View the HTML code in your Web page	Click the Source tab in the Web Page Designer.
Add controls to a Web page	Display the Web page in Design view, open the Toolbox (which automatically contains Visual Web Developer controls), position the insertion point where you want to place the control on the page, and then double-click the control in the Toolbox.
Change the name of an object on a Web page	Use the Properties window to change the object's *ID* property to a new name.
Write the default event procedure for an object on a Web page	Double-click the object to display the code-behind file, and write the event procedure code for the object in the Code Editor.
Verify the format of the data entered by the user into a control on a Web page	Use one or more validator controls from the Validation tab of the Toolbox to test the data entered in an input control.
Run and test a Web site in Visual Studio	Click the Start Debugging button on the Standard toolbar. Visual Studio builds the project and loads the Web site in Internet Explorer.
Create an HTML page for a project	Click the Add New Item command on the Web site menu, and then add the new HTML Page template to the project. Create and format the HTML page by using the Web Page Designer.
Create a link to other Web pages on your Web site	Add a *HyperLink* control to your Web page, and then set the control's *NavigateUrl* property to the address of the linked Web page.
Display database records on a Web page	Add a *GridView* control to a Web page in the Web Page Designer. Establish a connection to the database and format the data by using commands in the Common GridView Tasks list. (The Choose Data Source command starts the Data Source Configuration Wizard.)
Set the title displayed for Web pages on the Internet Explorer title bar	For each Web page, use the Properties window to set the *DOCUMENT* object's *Title* property.