HOUR 3

Using Visual Web Developer

In this hour, we will cover

- Creating new websites and web pages
- Opening existing websites
- Customizing the editor through Visual Web Developer's Options menu
- Using techniques for laying out HTML content through the Design view
- Moving and resizing the assorted windows
- Accessing help through the installed documentation

In the preceding hour we looked at the ASP.NET programming model, noting how ASP.NET pages are composed of an HTML portion and a source code portion. Recall that the HTML portion of an ASP.NET web page consists of static HTML markup and Web control syntax; the source code portion, separated out into its own file, is implemented as a class with various event handlers.

In the past two hours, you got a cursory look at Visual Web Developer, the development environment we'll be using throughout this book to build ASP.NET pages. In Hour 1, "Getting Started with ASP.NET 2.0," we installed Visual Web Developer and received a quick tour of its features. In the preceding hour we delved a bit deeper into Visual Web Developer's interface, looking at how to create a new ASP.NET website and how to use the WYSIWYG Design view, the HTML Source view, and the source code editor.

Because we'll be using Visual Web Developer extensively throughout this book, it behooves us to take a moment to fully explore this editor. Visual Web Developer is a sophisticated and powerful programming editor, with a bevy of features and capabilities. As with any profession, it's important to have a solid grasp of the tools at your disposal.

Creating a New Website

When you start Visual Web Developer, you will typically want to do one of two things: either create a new website or open an existing website. A website is a collection of resources: static and dynamic web pages, graphic files, style sheets, configuration files, and so on. In addition to various files, a website may contain subdirectories, each of which may contain its own set of files and further subdirectories. A website is akin to a folder on your personal computer: It's a repository for files and subfolders.

To create a new website with Visual Web Developer, go to the File menu and select New Web Site or simply click the New Web Site icon in the Toolbar. Either of these actions will bring up the New Web Site dialog box, shown in Figure 3.1.

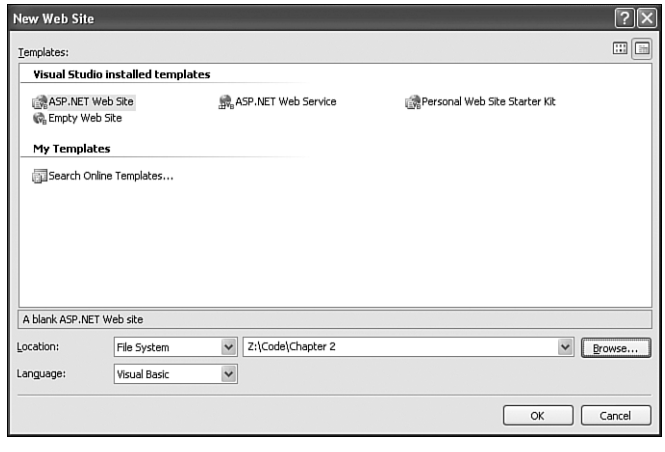

FIGURE 3.1 Create a new website from the New Web Site dialog box.

Choosing a Website Template

When creating a new website, you can choose among a number of available templates. For example, there's the ASP.NET Web Site template, the ASP.NET Web Service template, the Personal Web Site Starter Kit template, and an Empty Web Site template. Regardless of what template is selected, a website will be created; the differences among the templates is what default files the template includes with the website. For example, in the preceding hour we saw that creating a new website using the ASP.NET Web Site template creates a website with an `App_Data` folder and three files: `Default.aspx`, `Default.aspx.vb`, and `web.config`. Creating a website using the Empty Web Site template will create the website but will not add *any* default folders or files.

Did you Know?

The website templates are available to hasten the startup involved in creating a web application. The Personal Web Site Starter Kit template, for example, will build a website with a number of existing pages and features to help you in creating a personal website. (For more information on this template, see http://msdn.microsoft.com/asp.net/archive/default.aspx?pull=/library/enus/dnaspp/html/pws.asp.)

Be sure to try out the Search Online Templates feature in the My Templates section. This option, if selected, will look online for additional website templates created and distributed by Microsoft and others.

Although there is an array of website templates to choose from, all of the examples in this book will be created using the ASP.NET Web Site template.

Specifying the Website's Location

Websites can be located either on your personal computer or on a remote computer. Typically, personal computers do not double as web servers; that is, chances are the personal computer on which you're working right now does not host websites. Rather, you use your PC for your own ends—surfing the web, checking email, playing games, and so on. Publicly available websites are often hosted through **web-hosting companies**, which offer always-on computers with a persistent connection to the Internet. These computers' sole purpose is to host a website; they have web server software running on them and essentially sit around and wait for incoming requests. Upon receiving a request for a web page, they render the page and return the resulting markup to the requesting browser.

Often developers place a website on their own personal computer during the building and testing phases. The site won't be accessible over the Internet, but that's okay because the site is not yet ready for the public. When a functional site is complete, though, it can be moved to a remote web-hosting company so that the site can be accessed by anyone with an Internet connection. However, you can opt to create a new site on a remote computer from the get-go. All you need is an account with a web-hosting company.

Did you Know?

A gaggle of web-hosting companies is available with various features and pricing plans. You can find these web-hosting companies through any search engine or through sites such as HostIndex.com or TopHosts.com, which serve as a directories of web-hosting companies.

If you decide to host your ASP.NET website with a web-hosting company, be sure to check (and double-check) with the company to ensure that its servers support ASP.NET 2.0.

If you opt to host the website on your local computer you can host it in, potentially, one of two ways:

- **Through the file system**—With this approach, you provide a directory on your hard drive that serves as the website's **root directory**. All of the site's associated files and folders will be placed in that specified directory.
- **Through IIS, Microsoft's Web server**—If your personal computer has Internet Information Services (IIS) installed, you can host the website locally through IIS. IIS can be installed on Windows XP Professional and Windows 2003 Server; it is not able to be installed on Windows XP Home edition.

If you want to host the site locally, from the Location drop-down list, select File System. Next, click the Browse button to the right of the Location drop-down list. This will bring up the Choose Location dialog box shown in Figure 3.2.

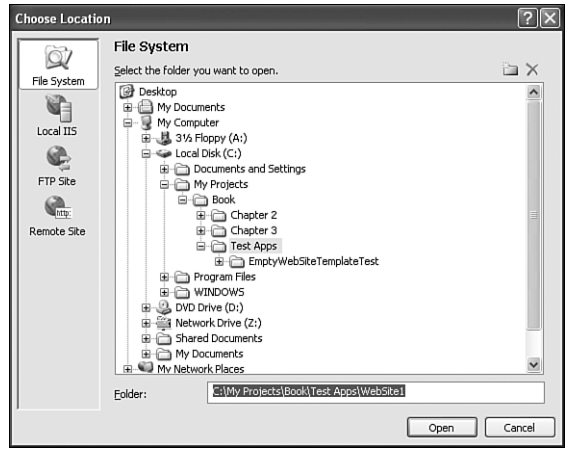

FIGURE 3.2 Choose the location where your website will reside.

The left column in the Choose Location dialog box lists the various locations the website can be saved.

By the Way

If your personal computer doesn't have Microsoft's IIS web server installed, you may be wondering how, in the preceding hour, we were able to view an ASP.NET page through a browser. This is possible because Visual Web Developer ships with a scaled-down web server referred to as the **ASP.NET Development Web Server.**

This web server is designed solely for testing websites locally and will refuse any attempted access from outside your own computer. It is shipped with Visual Web Developer, so those developers who run Windows XP Home, which does not support IIS, can still create, build, and test ASP.NET applications.

To host the website on a remote computer, select either the HTTP or FTP options for location and specify the HTTP or FTP address. For the FTP settings, you'll need to provide the FTP server, port, directory, and username/password, if anonymous access is not allowed. Similarly, if you choose to use the HTTP setting when attempting to create the website, you'll be prompted for a username and password. Contact your web-hosting company for information on whether you should use HTTP or FTP access and what settings you'll need to use in order to connect.

All of the websites throughout this book will be created on the local file system.

Choosing the Source Code Programming Language

The setting chosen in the Language drop-down list specifies the programming language of your ASP.NET web pages' source code portions. Two options are available: Visual Basic and Visual C#. As discussed in the preceding hour, the Visual Basic language will be used for the examples throughout this book.

By the Way

If you have a programming background in Java or C/C++, you may be more familiar with Visual C# than Visual Basic. Visual Basic is typically preferred by those with a background in the language or those new to programming.

If you are interested in learning more about Visual C#, check out Microsoft's Visual C# Developer Center (http://msdn.microsoft.com/vcsharp/) along with *Microsoft Visual C# 2005 Unleashed* by Kevin Hoffman (ISBN: 0672327767).

To gain practice creating websites in Visual Web Developer, go ahead and create a new website using the ASP.NET Web Site template, the File System location with the website in a directory of your choice, and with the Language set to Visual Basic. As we saw in the preceding hour, this will create a website with an `App_Data` folder, a `Default.aspx` ASP.NET page (with a corresponding `Default.aspx.vb` file), and a configuration file, `web.config`. These files are listed in the Solution Explorer, which you can find in the upper-right corner. (If you do not see the Solution Explorer, go to the View menu and choose the Solution Explorer option.)

Opening Existing Websites

Now that we've created a website, let's see how to open this website at a later point in time. First, go ahead and close the website. This can be accomplished by closing Visual Web Developer altogether, or by going to the File menu and choosing Close Project.

After closing your website, reopen it by going to the File menu and choose the Open Web Site menu option. This will list the Open Web Site dialog box. This dialog box is nearly identical to the Choose Location dialog box shown in Figure 3.2.

Because you created your website locally through your personal computer's file system, open the site by selecting the File System icon in the left column and then navigating to the folder where you placed the website. Finally, click the Open button to open the website.

Opening the website will close the existing opened website (if any) and load the selected website's contents into the Solution Explorer. At this point you can work with the website as you normally would, creating and editing web pages' HTML and source code portions.

Did you Know?

If you have recently worked with a particular website, there's a quicker way to open it with Visual Web Developer. The File menu contains a Recent Projects menu item that lists the most recently opened projects. Clicking on a project name from the Recent Projects list opens that project.

Working with Web Pages and Other Content

A website is simply a repository of related files and subdirectories. Websites typically contain files of the following types:

- **Static web pages**—An **HTML page** is a static web page. Unlike an ASP.NET page, it contains only HTML content—no Web controls and no source code. As its name implies, the content of these types of files is static, and cannot be altered based on user input, server-side data, or other criteria.
- **ASP.NET web pages**—ASP.NET pages are the dynamic web pages in your site. They are implemented as two files: *`PageName`*`.aspx`, which contains the HTML portion; and *`PageName`*`.aspx.vb`, which contains the source code portion.
- **Image files**—Most websites have various images, logos, and clip art. These image files typically are stored on the website, either in the root directory or in an Images subdirectory.
- **Configuration files**—ASP.NET websites contain a configuration file named `web.config`, which provides server-side setting information.
- **Style sheet files**—Style sheets are files that spell out display information. For example, in your site you might want all content to be displayed in the Arial

font and have content within <h1> tags displayed in italics. You can specify this aesthetic information through style sheet files. For more information on style sheets, refer to http://www.w3schools.com/css/.

- **Script files**—In addition to server-side source code, a web page may contain **client-side script code**. This is code that is sent to and runs on the end user's web browser. Often this script is packaged in a separate script file on the web server, which the browser requests as needed.

This list of file types enumerates the most commonly found file types on a web server but is hardly exhaustive. A rock band's website, for example, might also have MP3 files available for download. Additionally, numerous ASP.NET-specific files can optionally be added to your website to provide various types of functionality. We'll be learning about many of these different ASP.NET-specific file types throughout this book.

Adding Content to Your Website

When you create a new website using the ASP.NET Web Site template, the new website has the `web.config` file along with a single ASP.NET page, `Default.aspx` (which is really composed of two files, `Default.aspx` and `Default.aspx.vb`). You can easily add additional files and folders through the Solution Explorer. From the Solution Explorer, start by right-clicking on the website name; this will bring up the context menu shown in Figure 3.3.

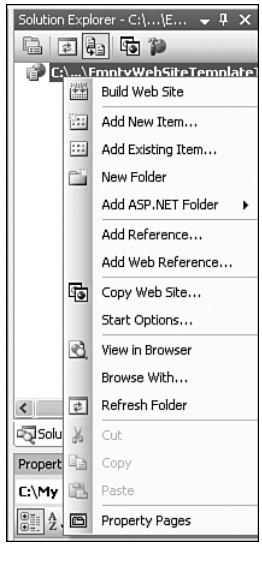

FIGURE 3.3
To add a new file or folder, right-click on the website name in the Solution Explorer.

To add a new folder to your website, select the New Folder item from the context menu. To add a new file, choose Add New Item. Selecting Add New Item will display the Add New Item dialog box (see Figure 3.4). The Add New Item dialog box lists the wide variety of types of files that can be added. Notice that there are file types for each of the popular file types enumerated earlier, in addition to many other types.

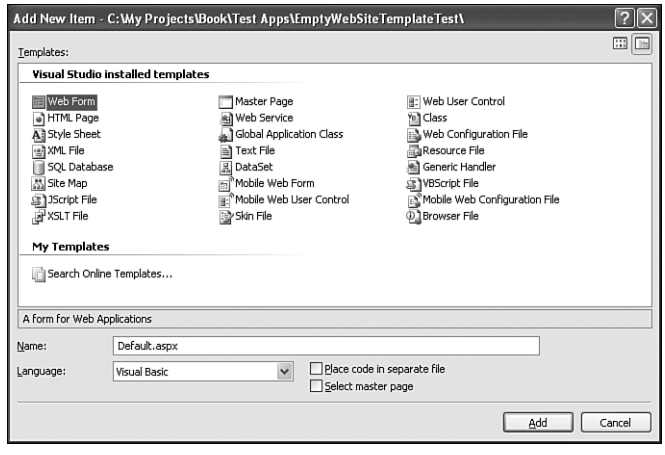

FIGURE 3.4
The Add New Item dialog box allows you to choose the type of file to add.

By the Way

To add a new ASP.NET page to your website, add an item of type Web Form.

At the bottom of the Add New Item dialog box, you'll find a series of options. The options displayed depend on what file type you have decided to add. For Web Forms, which are the item type name for ASP.NET pages, there are four options:

- **Name**—This indicates what the file will be named.
- **Language**—This dictates the language of the page's server-side source code portion.
- **Place Code in Separate File**—This specifies whether the source code portion should be implemented as a second file (*PageName*`.aspx.vb`) or if server-side `<script>` blocks will be used instead.
- **Select Master Page**—A **master page** is a site-wide template that can be applied to ASP.NET pages to maintain a consistent look and feel across the site. If you are using master pages, you can check this option to assign a master page to the newly created ASP.NET page.

By the Way

Master pages are a very useful way to create a consistent page layout across all pages in your site. We'll discuss the benefits of master pages, along with how to use them in your ASP.NET website, in Hour 21, "Using MasterPages to Provide Site-Wide Page Templates."

The Language drop-down list value for the ASP.NET page's source code portion will be the same language choice you specified when creating the website. However, a single ASP.NET website can have web pages that use different programming languages for their source code portions. However, I recommend against this approach and encourage you to stick with a single, unified programming language choice across all ASP.NET pages for a given website.

Although ASP.NET pages will work just as well if their source code portion is in the `.aspx` page in a server-side `<script>` block or if it is relegated to a separate file (*`PageName`*`.aspx.vb`), keep in mind that all of the examples we'll be working through in this book use the separate page model. Therefore, when adding a new ASP.NET page to your website, be sure to check the Place Source Code in a Separate File check box. Doing so will create both the *`PageName`*`.aspx` and *`PageName`*`.aspx.vb` files.

Watch Out!

The Place Code in Separate File option in the Add New Item dialog box is "sticky." That is, Visual Web Developer remembers your selection. Unfortunately, this "stickiness" is not remembered across projects. That is, if you create a new ASP.NET website project, Visual Web Developer will revert back to the default—to have this option unchecked. Therefore, whenever adding a new ASP.NET page, take a quick moment to ensure that this check box is indeed checked.

Let's practice adding a new ASP.NET page to our website. Imagine that in addition to `Default.aspx`, we also want to have a second ASP.NET page, `DisplayTime.aspx`. To add this page to your website, perform the following steps:

1. Go to the Solution Explorer and right-click on the website name.
2. Choose Add New Item from the context menu.
3. From the Add New Item dialog box (see Figure 3.4), select to add an item of type Web Form.
4. Enter **`DisplayTime.aspx`** for the page's Name, leave the Language setting as Visual Basic, and check the Place Source Code in a Separate File check box.
5. Click the Add button to create the new ASP.NET page.

You can follow these same steps to add other types of resources to your website. Of course, the options present in step 4 will differ depending on the type of item being added.

Adding Existing Content

Along with adding new content to your website, you can use Visual Web Developer to easily add existing content. You may already have an image file on your hard drive or an ASP.NET page from another project that you want to include in this project as well. If that's the case, you can add an existing item by right-clicking on the website name in the Solution Explorer and choosing Add Existing Item.

Choosing this option will display the standard file browsing dialog box. From here, you can navigate to the folder on your hard drive that contains the content you want to add, select it, and click the Add button. This will copy over the selected item to your website's directory, making it part of your website now.

Moving, Renaming, and Deleting Content

Along with adding new folders and files, from the Solution Explorer you can also move and delete content. To move content among the folders in your website, simply drag the file or folder from its existing location to a new file or folder.

To rename or delete a file or folder, start by right-clicking on the item in the Solution Explorer. This will bring up the context menu shown in Figure 3.5. As you can see from the figure, Rename and Delete menu items are available. Simply click on the appropriate menu item to rename or remove the selected file or folder.

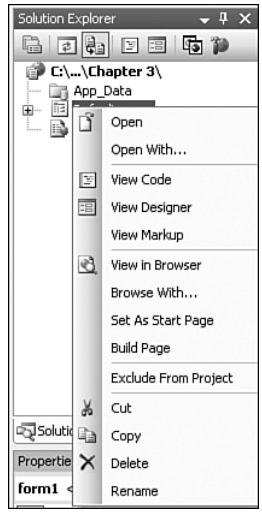

FIGURE 3.5 Select the appropriate menu item from the context menu.

Customizing the Visual Web Developer Experience

Like any robust programming editor, Visual Web Developer is highly customizable, enabling developers to configure the editor in a way that maximizes their productivity. Not only does Visual Web Developer give you fine-grained control over a variety of settings, but it also provides an easy way to export your unique settings to a single file. You can then re-create your environment at a new computer by importing your settings file. This makes it easy to move your settings from a desktop computer to a laptop; additionally, if you place your settings file on a website or keep it saved on a USB drive or web-based email account, you can easily re-create your development environment at *any* computer where you end up working!

In this section we'll examine how to customize Visual Web Developer. The bulk of the customizability is accomplished through the Options dialog box, which is available through the Tools menu. There are literally hundreds of settings, so we won't have the time to go through each and every one. Instead, we'll focus on the more germane settings. We'll also see how to alter the Visual Web Developer panes and their display settings, along with settings that aid with laying out HTML content in the Design view.

By default, the Visual Web Developer Options dialog box shows only the most pertinent options. To see all available options, check the Show All Settings check box.

By the Way

The screenshots throughout this book have been taken using the default Visual Web Developer settings. If you customize the environment to suit your preferences, there may be some disparity between your screen and the screenshots in this book.

Examining the Environment Settings

The majority of the customizable settings in Visual Web Developer are accessible via the Options dialog box, which you can display by selecting the Options item from the Tools menu. Figure 3.6 shows the Options dialog box when first opened.

The Options dialog box is broken down into various hierarchical categories, which are listed in the left column. Selecting an item from the list on the left displays its corresponding options on the right. As Figure 3.6 shows, the default category selected when opening the Options dialog box is the Environment category.

There are far too many categories and options available in this dialog box to exhaustively list them all; instead, let's just focus on the more pertinent ones.

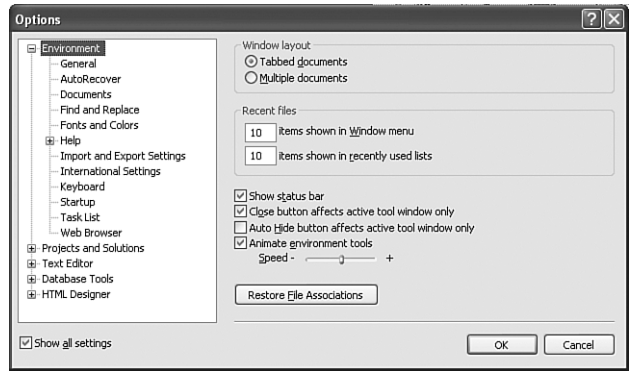

FIGURE 3.6 Customize your Visual Web Developer experience through the Options dialog box.

The Environment has two settings worth exploring. The first is the AutoRecover setting. While you are working on a website, Visual Web Developer will automatically save copies of modified files every so often, based on a setting you provide (the default is every five minutes). These backup copies are kept around for a specified number of days (seven, by default). The AutoRecover feature protects against losing hours of work due to an unexpected shutdown. If your officemate LeRoy happens to walk past your desk and kick the power cord out of the wall, in the worst case you'll not have the last five minutes of changes saved (assuming you left the AutoRecover frequency as the default). If you have AutoRecover enabled (which it is, by default), backup copies of your modified files will be saved periodically to the `\My Documents\Visual Studio 2005\Backup Files\<projectname>` folder.

The final Environment setting we'll look at is the Fonts and Colors settings, which is shown in Figure 3.7. The Fonts and Colors settings dictate the fonts, sizes, and colors of the text used in Visual Web Developer. You can alter the fonts and colors for a variety of settings: the Visual Web Developer text editor, the printed output, various debugging windows, and so on. Furthermore, for each variety of setting, there are multiple display items whose font and color can be customized. For example, the default fonts and colors for the Text Editor setting's Plain Text is black, 10pt, Courier New; its setting for the Selected Text is white with a blue background, 10pt, Courier New.

Many developers tweak the Fonts and Colors settings to make it easier to see certain types of tokens in their source code. For example, in the version of Visual Web Developer I use, I have Numbers displayed as purple, bold, 10pt, Courier New text, and Strings displayed as turquoise, 10pt, Courier New text.

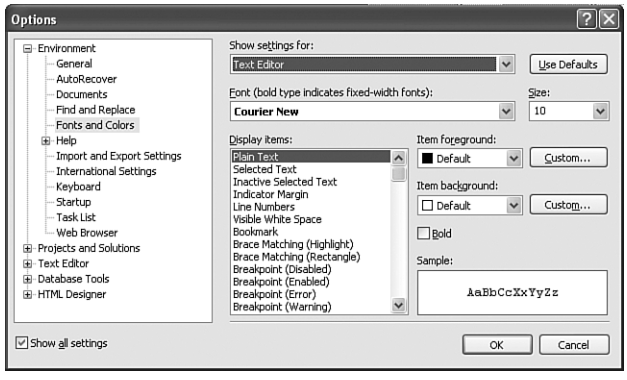

FIGURE 3.7 Specify the fonts, sizes, and colors of the text used in Visual Web Developer.

Configuring the HTML Design-Time Experience

The last settings item in the Options dialog box worth noting is the HTML Designer category. Expand this category and select the General item beneath it. On the right you should see an option titled Start Pages In with two options: Source View and Design View. Recall that when editing an ASP.NET page's source code portion, you can either work with the HTML content directly, through the Source view, or drag and drop HTML elements and Web controls onto the page from the Design view. By default, when you're creating a new web page or opening an existing one, the page is opened in the Source view. If you prefer the Design view, you can change the default behavior here.

The second setting in the HTML Designer category that merits discussion is the CSS Positioning item (see Figure 3.8). The key setting in this item is the Positioning options. If you check the first check box—Change Positioning to the Following for Controls Added Using the Toolbox, Paste, or Drag and Drop—you can indicate how items should be laid out on the designer. By default, this check box is unchecked, meaning that no positioning information is applied to Web controls or HTML elements added through the Design view. In the preceding hour we touched on this, discussing how extraneous whitespace is ignored in HTML, and to position elements, you need to use `<table>` elements like we did in the example in Hour 2, "Understanding the ASP.NET Programming Model," or other techniques that we have yet to explore.

One such positioning technique is **absolute positioning**. This technique gives each HTML element and Web control dragged onto the designer a fixed coordinate for its upper-left corner. These coordinates are sent down to the browser and are positioned accordingly. If you check the Change Positioning to the Following for Controls Added Using the Toolbox, Paste, or Drag and Drop check box and select the

Absolutely Positioned item from the drop-down list, you will be able to simply drag Web controls and HTML elements to the position you want on the designer.

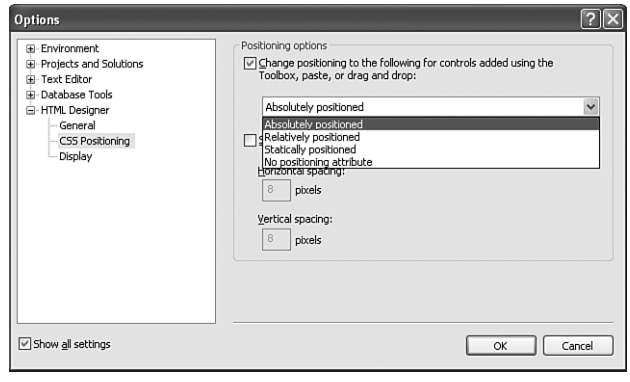

FIGURE 3.8
Control the HTML layout to a finer degree with absolute positioning.

Figure 3.9 illustrates this by examining the designer after configuring Visual Web Developer to use Absolutely Positioned elements. Specifically, I have added a Button Web control and then dragged it near the middle of the design surface. More precisely, the absolute position of the Button Web control is 119 pixels from the top and 100 pixels from the left, as can be seen in the lower-left corner of the Visual Web Developer status bar.

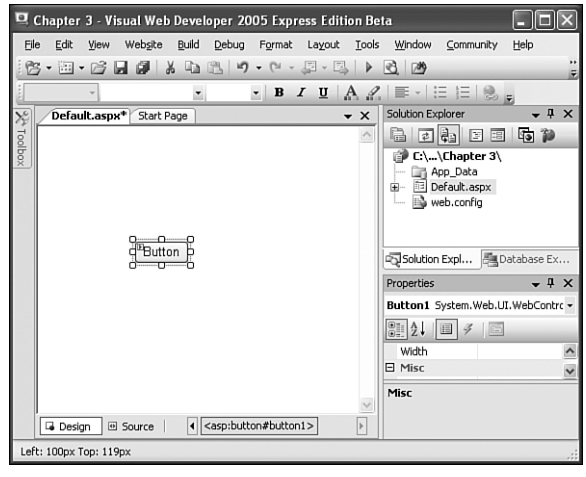

FIGURE 3.9
A button has been absolutely positioned to 119px from the top and 100px from the left.

While absolute positioned elements may seem like a godsend, making it supremely simple to move elements around to specific locations, it is important to be aware of the risks. For starters, when you position elements absolutely, you are making inherent assumptions about your visitors' screen resolutions. For example, imagine that

you are designing your site on a monitor with a resolution that's 1,024 pixels wide and 768 pixels tall and you want a Button Web control to be placed near the right side of the page. You may drag the Button to have an absolute position of, say, 900 pixels from the left and 100 pixels from the top.

When a visitor reaches your site, his browser will place that Button 900 pixels from its left margin and 100 pixels from its top. But what if the visitor's monitor supports only a resolution of 800 pixels wide by 600 pixels tall? That Button you meant to be on the right side of the screen is now off the visitor's screen, so he needs to scroll to the right to see it.

Another disadvantage of absolutely positioned elements is that they do not flow nicely as their size increases. When we reach Hour 15, "Displaying Data with the Data Web Controls," we'll look at a Web control that displays database data. Since at design time we might not know how much data will be displayed (because that would depend on how much data is in the database, which will change over time), we may run into problems if we put a GridView on our page and, say, a Button beneath it. When there are but a few items in the GridView, the Button may be positioned nicely, but as time goes on and more data is added to the database, the GridView's dimensions will increase and eventually its larger size will cover up the Button beneath it.

I am not categorically saying that you should never use absolute positioned elements. They make perfect sense in a controlled environment, like an intranet, where you can be assured that your visitors all have a certain minimum resolution so you can design the site accordingly. However, don't simply opt to use absolute positioned elements because it seems simpler than learning the more intricate methods of positioning using HTML syntax. The time you save in bypassing learning positioning with HTML syntax is likely time that will be taken back from you when some of these common absolute position design issues arise.

By the Way

To learn more about positioning HTML content without using absolute positioning, refer to http://www.w3schools.com/html/html_layout.asp.

Did you Know?

When you have settled on the particular settings you find most conducive, take a moment to export your savings to a file. To accomplish this, go to the Tools menu and select the Import and Export Settings option. This will take you through a wizard where you can indicate what settings to export along with the filename and location to save the settings.

Once you have exported your settings, you can re-create your personalized settings on another computer by importing this settings file. This capability is useful if you develop on both a desktop and laptop, depending on whether you're onsite or not, or if you are a contractor who moves jobs every few months but want to maintain a consistent collection of settings.

Viewing, Moving, and Resizing Windows

The Visual Web Developer environment is made up of a bevy of windows that provide various tidbits of information. The window we've examined in most detail is the Solution Explorer, which lists the files and folders in the website. In the preceding hour, we saw two other windows: the Properties window, which listed the properties for the selected HTML element or Web control; and the Toolbox window, which listed the Web controls and HTML elements that could be dragged onto an ASP.NET page.

These are but three of the many windows available in Visual Web Developer. You can see a complete list of the available windows by going to the View menu.

Windows in Visual Web Developer each have a default position, size, and behavior. The position indicates where on the screen the window is placed and whether or not it is floating or docked. The Solution Explorer, for example, is in a docked position in the upper-right corner, by default; the Toolbox can be found in a docked position on the left. A docked window is one that's attached to a margin of the editor; when a docked window is shown, the content it displaces is moved elsewhere on the screen. A floating window is *not* attached to any margin; it floats above all other windows and content in the editor, covering it up rather than displacing it.

Each window also has a size. You can move your mouse to the margin of the window and click and drag to increase or decrease the window's width or height. Lastly, each window has a behavior: It's either pinned or unpinned (the unpinned behavior is sometimes referred to as Auto-Hide behavior). A pinned window remains displayed regardless of whether your mouse is over the window. An unpinned window is displayed when you move your mouse over the window and disappears when your mouse leaves the window's focus. You can toggle a window's pinned status by clicking the pin icon in each window's upper-right corner. Typically, I keep the Solution Explorer, Properties, and Toolbox windows pinned because they are commonly used; all other windows I'll make unpinned so they do not encroach on my screen's real estate.

In addition to being able to pin and unpin windows, you can also close a window. To remove a window from the screen, simply click the X icon in the upper-right corner of the window (it's next to the pin icon).

Did you Know?

If you cannot find a window onscreen where you expect it, you may have moved it or accidentally closed it. In any event, you can display the needed window by going to the View menu and selecting the appropriate menu option.

If you do not like the position of a window, you can easily move it. Simply click on the top of the window and, holding down your mouse button, drag your mouse to the location you want the window to appear.

A World of Help at Your Fingertips

ASP.NET is a rich, robust web development technology built on a platform known as the .NET Framework. This platform consists of hundreds of classes that provide the core functionality of the ASP.NET engine. Needless to say, it can take *years* to have a deep understanding of the framework and its capabilities.

Fortunately, Visual Web Developer provides a variety of documentation and help. The version of Visual Web Developer you have installed on your computer includes the MSDN Library for Visual Studio Express editions. The MSDN Library is Microsoft's colossal collection of articles, whitepapers, technical documentation, knowledge base content, and frequently asked questions and answers. To view the library, simply go to the Help menu and choose Search, Contents, or Index. This will launch the Microsoft Visual Studio 2005 Express Editions Documentation program, from which you can poke through the help.

Another neat feature of Visual Web Developer is its Dynamic Help. From the Help menu, select the Dynamic Help option. This will display the Dynamic Help window (which you can resize, position, and pin just like any other window). As its name implies, the Dynamic Help window shows context-sensitive help based on where your cursor is in the source code or HTML portions. For example, if you're in the Design view and you click on a Button Web control in your page, the Dynamic Help window will automatically display help links with titles like

- Button Web Server Control Overview
- How to: Add Button Web Controls to a Web Forms Page
- How to: Add ImageButton Web Controls to a Web Forms Page

There's also the Community menu, which contains menu items like Ask a Question and Check Question Status. These menu options plug into Microsoft's online forum site, which you can access directly through a web browser by going to http://forums.microsoft.com/msdn/.

In addition to the Microsoft online forum site, the Community menu item has links to other developer resource sites and tools to assist in searching online for answers, templates, samples, and controls.

Did you Know?

Although a wealth of information is available through the documentation installed on your computer and through Visual Web Developer's Community and Help menus, a vast array of information is also available online. One tip to help your searches is Google's Microsoft-specific search, available at http://www.google.com/microsoft.html. This customized search page returns only results relating to Microsoft technologies and topics, which can help expedite your search for answers to your questions.

Summary

We spent this hour investigating Visual Web Developer in greater detail. It is important that you have a familiarity with this editor since it is what you'll be using throughout this book and beyond. While we will have ample opportunity to sharpen our Visual Web Developer skills throughout the future hours, I thought it worthwhile to take some time to more formally explore the tool.

Specifically, we looked at how to create and open websites in Visual Web Developer. Websites can be located either locally, on your personal computer, or remotely, with a web-hosting company. Visual Web Developer makes it easy to work with both sites locally and remotely, and even includes a lightweight, built-in web server that allows for developers using Windows XP Home to run ASP.NET websites locally. After you have created a website, you'll want to add various files and folders. This is easily accomplished through the Solution Explorer.

This hour we also looked at customizing the environment through the Options dialog box and by repositioning and resizing the many windows. We concluded with a quick synopsis of Visual Web Developer's extensive built-in help system. The Dynamic Help capabilities are especially useful for developers new to ASP.NET.

In this hour and the past two we've covered a lot of ground. We've looked at the fundamentals of ASP.NET and the .NET Framework; installed the .NET Framework, Visual Web Developer, and SQL Server 2005 Express Edition; dissected the ASP.NET programming model; created our first ASP.NET web page, complete with HTML markup, Web controls, and server-side source code; and explored Visual Web Developer. We're now ready to build a nontrivial ASP.NET web page, which we'll tackle in the upcoming hour. This exercise will help hammer home many of the key points mentioned through these first three hours.

Q&A

Q. ***I want to use a web-hosting company to host my ASP.NET website. How do I determine whether a particular web-hosting company can host my ASP.NET application?***

A. The easiest way to ascertain whether a given web-hosting company can host your site is to simply ask. Be sure to tell the company that you are creating an *ASP.NET 2.0* website using Visual Web Developer. For your ASP.NET site to be able to run on the company's servers, the web-hosting company will need to be running Microsoft's web server, IIS, along with the .NET Framework version 2.0. If these conditions are met, your site should run just fine.

Q. ***I have created a website on my local file system with the website located at folder X. I'd like to move the website to folder Y. Is this possible?***

A. Sure. To move the site, simply close Visual Web Developer and then move the website's folder from its current location to wherever else you'd like it to reside. After you have moved the files, reopen Visual Web Developer, go to the File menu, and choose the Open Web Site option. Browse to the *new* folder location and click Open. That's all there is to it!

Q. ***What are some online resources for ASP.NET information?***

A. There are a plethora of ASP.NET resources available free online. Your first destination should be the official ASP.NET website, www.asp.net. This site has links to virtually every ASP.NET resource site on the Net, along with a very active online forum site (http://forums.asp.net/) that has received more than 1,000,000 posts from hundreds of thousands of users.

Many of the larger ASP.NET resource sites have been around for many years, having started when there was no ASP.NET, only ASP. I run a popular online resource, 4GuysFromRolla.com, which has a messageboard, FAQs, and thousands of ASP/ASP.NET-related articles. Some other large and prominent ASP.NET sites include 15Seconds.com, ASPAlliance.com, and ASP101.com. There's also Microsoft's ASP.NET Developer Center, available at http://msdn.microsoft.com/asp.net/.

Workshop

Quiz

1. There are two local and two remote techniques through which a website can be created or opened. What are these four techniques?
2. True or False: IIS stands for Internet Information Service and is Microsoft's web server software.
3. True or False: The ASP.NET Development Web Server and IIS are the same thing.
4. When you're adding a new Web Form (ASP.NET page) to your website, there's a check box titled Place Source Code in a Separate File. How will the created ASP.NET page differ if you don't check this option versus if you do?
5. What are the advantages and disadvantages of having absolute positioned Web controls and HTML elements?

Answers

1. The two local techniques are through the local file system or through a local version of IIS, Microsoft's web server software. (Keep in mind that in order for you to use the local IIS option, your personal computer must have IIS installed. IIS is not installable on Windows XP Home edition, and, even if your operating system supports IIS, it might not currently be installed.) The two remote techniques are through HTTP or FTP. If you are using a web-hosting company, talk to the company to determine which approach to use and the various settings to use.
2. True.
3. False. IIS is Microsoft's professional grade web server software. ASP.NET Development Web Server is a lightweight web server that ships with Visual Web Developer to enable those who do not have IIS installed to be able to still develop, build, and test ASP.NET applications. IIS is designed to run real-world websites; the ASP.NET Development Web Server is designed solely for local-only, low-stress testing.
4. If you do *not* check the Place Source Code in a Separate File check box, only one file will be created for the ASP.NET page, *`PageName`*`.aspx`. Your source code portion will need to be placed within a server-side `<script>` block. Preferably, you'll check the Place Source Code in a Separate File check box, in

which case *two* files will be created: *PageName*.aspx and *PageName*.aspx.vb. In this scenario, the HTML markup and Web control syntax resides in the former file, while the source code portion resides in the latter.

5. Absolute positioned elements have the advantage that they're easy to place on a page: You simply drag and drop from the Design view. The disadvantage is that using absolute positioning makes implicit assumptions about the end user's screen resolution. Since you, as the developer, are laying out the site according to your screen's resolution, the results may look good for visitors who use a similar resolution but will likely look "off" for those visitors who use a much lower or higher resolution.

   Additionally, absolute positioned elements that can grow dynamically based on the data bound to them have an annoying knack of consuming more screen real estate than originally estimated, thereby potentially overlapping any elements beneath it.

Exercises

1. For more practice with Visual Web Developer, take a moment to create a new website. Along with the Default.aspx page, add some additional ASP.NET pages. In each page, add various content through the Design view or Source view, much like we did in the preceding hour. Tinker around with the editor, not worrying about whether you're doing things right or wrong. Just experiment.

   After creating a couple of pages, try marking each page as the start page (right-click on the ASP.NET page in the Solution Explorer and choose Set as Start Page) and then visit the page in a browser by going to the Debug menu and choosing Start Without Debugging.

   If you're feeling adventurous, try adding some image files from your hard drive to your website. Next, have them displayed in your ASP.NET pages using Image Web controls, much like we did in the preceding hour. Again, focus more on the exploration than whether you are doing things the right way. Take your time and have fun; it's the best way to learn!

This page intentionally left blank