Scott Mitchell

STARTER KIT

DVD includes Microsoft Visual Web Developer 2008 Express Edition

Sams Teach Yourself

ASP.NET 3.5

in 24 Hours

SAMS

Sams Teach Yourself ASP.NET 3.5 in 24 Hours, Complete Starter Kit

Copyright © 2008 by Pearson Education, Inc.

All rights reserved. No part of this book shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from the publisher. No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained herein.

ISBN-13: 978-0-672-32997-5
ISBN-10: 0-672-32997-2

Library of Congress Cataloging-in-Publication Data:

Mitchell, Scott.
Sams teach yourself ASP.NET 3.5 in 24 hours : complete starter kit / Scott Mitchell.
p. cm.
ISBN 0-672-32997-2
1. Active server pages. 2. Web sites—Design. 3. Microsoft .NET. I. Title.
TK5105.8885.A26M587 2008
005.2'76—dc22
2008014770

Printed in the United States of America

First Printing June 2008

Trademarks

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Sams Publishing cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Warning and Disclaimer

Every effort has been made to make this book as complete and accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The author and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the CD or programs accompanying it.

Bulk Sales

Sams Publishing offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales. For more information, please contact

U.S. Corporate and Government Sales
1-800-382-3419
corpsales@pearsontechgroup.com

For sales outside the U.S., please contact

International Sales
international@pearson.com

Editor-in-Chief
Karen Gettman

Executive Editor
Neil Rowe

Development Editor
Mark Renfrow

Managing Editor
Kristy Hart

Project Editor
Andrew Beaster

Copy Editor
Barbara Hacha

Indexer
Erika Millen

Proofreader
Kathy Ruiz

Technical Editor
Todd Meister

Publishing Coordinator
Cindy Teeters

Multimedia Developer
Dan Scherf

Book Designer
Gary Adair

Composition
Nonie Ratcliff

This Book Is Safari Enabled

The Safari® Enabled icon on the cover of your favorite technology book means the book is available through Safari Bookshelf. When you buy this book, you get free access to the online edition for 45 days.

Safari Bookshelf is an electronic reference library that lets you easily search thousands of technical books, find code samples, download chapters, and access technical information whenever and wherever you need it.

To gain 45-day Safari Enabled access to this book:

- Go to www.informit.com/onlineedition.
- Complete the brief registration form.
- Enter the coupon code 24DQ-UXWD-LCKF-B9DS-H2C9.

If you have difficulty registering on Safari Bookshelf or accessing the online edition, please email customer-service@safaribooksonline.com.

Introduction

As the World Wide Web continues its meteoric growth, websites have matured from simple collections of static HTML pages to data-driven dynamic web applications. For example, websites such as eBay or Amazon.com are much more than a collection of HTML pages—they are complex applications that happen to be accessed through the Internet. Although many competing technologies exist for building data-driven websites, this book shows how to use the latest version of Microsoft's popular ASP.NET technology for creating web applications.

ASP.NET web applications are composed of individual ASP.NET web pages. As we will see in numerous examples throughout this book, these ASP.NET pages can display HTML, collect user input, and interact with databases. ASP.NET pages contain a mix of both HTML and source code. It is the source code of an ASP.NET page that allows for the more advanced features, such as accessing data from a database, or sending an email. The source code of an ASP.NET web page can be written in any one of a number of programming languages. For this book we will be using Microsoft's Visual Basic programming language. Don't worry if you've never programmed in Visual Basic, or even if you have never programmed at all. Starting with Hour 5, "Understanding Visual Basic's Variables and Operators," we spend three hours examining programming language concepts and the Visual Basic syntax.

To ease ASP.NET web page development, Microsoft provides a free development editor, Visual Web Developer, which is included in this book's accompanying CD. Visual Web Developer simplifies creating both the HTML and source code portions of ASP.NET pages. The HTML for an ASP.NET web page can be quickly created by using the Designer, which is a What You See Is What You Get (WYSIWYG) graphical editor. With the Designer, you can drag and drop various HTML elements onto an ASP.NET web page, moving them around with a few clicks of the mouse. Likewise, Visual Web Developer offers tools and shortcuts that help with creating an ASP.NET page's code.

Audience and Organization

This book is geared for developers new to ASP.NET, whether or not you've had past experience with HTML or programming languages. By the end of this book you'll be able to create and deploy your own dynamic, data-driven web applications using ASP.NET.

This book's 24 hours are divided into four parts. Part I introduces you to ASP.NET, HTML, Visual Web Developer, and Visual Basic. Hour 1, "Getting Started with ASP.NET 3.5," begins with an overview of ASP.NET and then walks you through installing the .NET Framework, Visual Web Developer, and other necessary components. Hour 3, "Using Visual Web Developer," showcases Visual Web Developer, which is the powerful development editor you'll be using throughout this book to create ASP.NET web pages. Hours 5, 6, and 7 examine the syntax and semantics of the Visual Basic programming language.

ASP.NET offers a variety of user interface elements for collecting user input, including text boxes, check boxes, drop-down lists, and radio buttons. In Part II you will see how to collect and process user input. Hour 10, "Using Text Boxes to Collect Input," examines using single-line, multi-line, and password text boxes, while Hour 11, "Collecting Input Using Drop-Down Lists, Radio Buttons, and Check Boxes," examines alternative user input controls.

Part III shows how easy it is to build data-driven websites with ASP.NET. Starting in Hour 13, "An Introduction to Databases," we begin our look at building websites that interact with databases. Typically, data-driven websites enable visitors to view, update, delete, and insert data into the database from an ASP.NET page. In Hour 15, "Displaying Data with the Data Web Controls," you will learn how to display database data in a web page. Hour 16, "Deleting, Inserting, and Editing Data," examines how to edit, insert, and delete data.

Part IV highlights tools provided by ASP.NET and Visual Web Developer that help with building professional, easy-to-use websites. In Hour 20, "Defining a Site Map and Providing Site Navigation," you'll see how to define a website's navigational structure and display menus, treeviews, and breadcrumbs. Hour 22, "Using Master Pages to Provide Sitewide Page Templates," examines master pages, which enable web designers to create a web page template that can be applied to all pages across the site.

Conventions Used in This Book

This book uses several design elements and conventions to help you prioritize and reference the information it contains:

By the Way

By the Way boxes provide useful sidebar information that you can read immediately or circle back to without losing the flow of the topic at hand.

Did You Know? boxes highlight information that can make your Visual Basic programming more effective.

Did you Know?

Watch Out! boxes focus your attention on problems or side effects that can occur in specific situations.

Watch Out!

New terms appear in a **semibold** typeface for emphasis.

In addition, this book uses various typefaces to help you distinguish code from regular English. Code is presented in a `monospace` font. Placeholders—words or characters that represent the real words or characters you would type in code—appear in *`italic monospace`*. When you are asked to type or enter text, that text appears in **`bold monospace`**. Menu options are separated by a comma. For example, when you should open the File menu and choose the New Project menu option, the text says "Select File, New Project."

Some code statements presented in this book are too long to appear on a single line. In these cases, a line-continuation character ➥ is used to indicate that the following line is a continuation of the current statement. Furthermore, some code listings include line numbers. These numbers are used to refer to specific lines of code in the text and are not part of the code syntax.

I hope you enjoy reading this book as much as I enjoyed writing it.

Happy Programming!

Scott Mitchell
mitchell@4guysfromrolla.com

HOUR 1

Getting Started with ASP.NET 3.5

In this hour, we will cover

- What is ASP.NET?
- System requirements for using ASP.NET
- Software that must be installed prior to using ASP.NET
- Installing the .NET Framework, Visual Web Developer, and SQL Server 2005
- Taking a quick tour of Visual Web Developer
- Creating a simple ASP.NET web page and viewing it through a web browser

ASP.NET is an exciting web programming technology pioneered by Microsoft that allows developers to create **dynamic web pages**. Dynamic web pages are pages whose content is dynamically regenerated each time the web page is requested. For example, after you log on, the front page of Amazon.com shows books it recommends for you, based on your previous purchases. This is a dynamic web page because it is a single web page whose content is customized based on what customer is visiting. In this book we examine how to create dynamic ASP.NET websites quickly and easily.

Prior to ASP.NET, Microsoft's dynamic web programming technology was called Active Server Pages, or ASP. Although ASP was a popular choice for creating dynamic websites, it lacked important features found in other programming technologies. Microsoft remedied ASP's shortcomings with ASP.NET. ASP.NET version 1.0 was released in January 2002 and quickly became the web programming technology of choice for many. In November 2005, Microsoft released the much-anticipated version 2.0. Two years later, in November 2007, Microsoft released ASP.NET version 3.5.

Before we can start creating our first ASP.NET website, we need to install the .NET Framework, Visual Web Developer, and SQL Server 2005. The .NET Framework is a rich platform for creating Windows-based applications and is the underlying technology used to create ASP.NET websites.

Visual Web Developer is a sophisticated program for creating, editing, and testing ASP.NET websites and web pages. ASP.NET web pages are simple text files, so any text editor will suffice (such as Microsoft Notepad), but if you've created websites before, you know that using tools such as Microsoft FrontPage or Adobe Dreamweaver makes the development process much easier than using a generic text editor like Notepad. This is the case for ASP.NET, as well.

The third and final piece we'll need to install is SQL Server 2005. SQL Server is a database engine, which is a specialized application designed to efficiently store and query data. Many websites interact with databases; any e-commerce website, for example, displays product information and records purchase orders in a database. Starting with Hour 13, "An Introduction to Databases," we'll see how to create, query, and modify databases through both Visual Web Developer and ASP.NET pages.

This hour focuses on getting everything set up properly so that we can start creating ASP.NET web applications. Although it would be nice to be able to jump straight into creating ASP.NET pages, it is important that we first take the time to ensure that the pieces required for ASP.NET are correctly installed and configured. We create a very simple ASP.NET page at the end of this hour, but we won't explore it in any detail. We look at ASP.NET pages in more detail in the next hour and in Hour 4, "Designing, Creating, and Testing ASP.NET Pages."

What Is ASP.NET?

Have you ever wondered how dynamic websites like Amazon.com work behind the scenes? As a shopper at Amazon.com, you are shown a particular web page, but the web page's content is dynamic, based on your preferences and actions. For instance, if you have an account with Amazon.com, when you visit Amazon.com's home page your name is shown at the top and a list of personal recommendations is presented further down the page. When you type an author's name, a title, or a keyword into the search text box, a list of matching books appears. When you click a particular book's title, you are shown the book's details along with comments and ratings from other users. When you add the book to your shopping cart and check out, you are prompted for a credit card number, which is then billed.

Web pages in websites whose content is determined dynamically based on user input or other information are called **dynamic web pages**. Any website's search engine page is an example of a dynamic web page because the content of the search results page is based on the search criteria the user entered and the searchable documents on the web server. Another example is Amazon.com's personal recommendations. The books and products that Amazon.com suggests when you visit the home page are different from the books and products suggested for someone else. Specifically, the recommendations are determined by the products you have previously viewed and purchased.

The opposite of a dynamic web page is a **static web page**. Static web pages contain content that does not change based on who visits the page or other external factors. HTML pages, for example, are static web pages. Consider an HTML page on a website with the following markup:

```
<html>
<body>
  <b>Hello, World!</b>
</body>
</html>
```

Such a page is considered a static web page because regardless of who views the page or what external factors might exist, the output will always be the same: the text `Hello, World!` displayed in a bold font. The only time the content of a static web page changes is when someone edits and saves the page, overwriting the old version.

By the Way

Virtually all websites today contain a mix of static and dynamic web pages. Rarely will you find a website that has just static web pages, because such pages are so limited in their functionality.

By learning ASP.NET, you will learn how to create websites that contain dynamic web pages. It is important to understand the differences between how a website serves static web pages versus dynamic web pages.

By the Way

Competing Web Programming Technologies

ASP.NET is only one of many technologies that can be employed to generate dynamic web pages. ASP.NET is the successor to Active Server Pages (ASP), which was Microsoft's earlier dynamic web-page creation technology. Other technologies include PHP, JSP, and ColdFusion.

By the Way

Personally, I find ASP.NET to be the easiest and most powerful technology of the bunch, which is why I'm writing a book about ASP.NET instead of one of the competing technologies. Moreover, the features and functionality of ASP.NET are head and shoulders above ASP. If you've created ASP pages in the past, you'll no doubt find that you can do the same things with ASP.NET but in a fraction of the time.

Did you Know?

If you have experience developing web applications with other web programming technologies, such as ASP, PHP, or JSP, you may already be well versed in the material covered in the next three sections. If this is the case, feel free to skip to the "Installing the ASP.NET Engine, Editor, and Database System" section.

Serving Static Web Pages

If you've developed websites before, you likely know that a website requires a **web server**.

A web server is a software application that continually waits for incoming **web requests**, which are requests for a particular URL (see Figure 1.1). The web server examines the requested URL, locates the appropriate file, and then sends this file back to the client that made the web request.

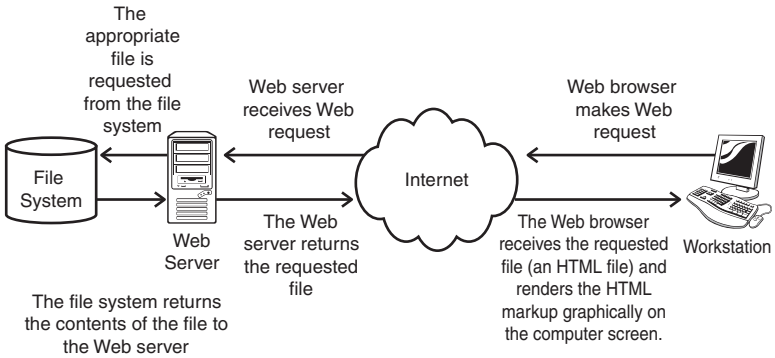

FIGURE 1.1
The web server handles incoming web requests.

For example, when you visit Amazon.com, your browser makes a web request to Amazon.com's web server for a particular URL, say `/books/index.html`. Amazon.com's web server determines what file corresponds to the requested URL. It then returns the contents of this file to your browser.

This model is adequate for serving static pages, whose contents do not change. However, such a simple model is insufficient for serving dynamic pages because the

web server merely returns the contents of the requested URL to the browser that initiated the request. The contents of the requested URL are not modified in any way by the web server based on external inputs.

Serving Dynamic Web Pages

With static web pages, the contents of the web page are just HTML elements that describe how the page should be rendered in the user's web browser. Therefore, when a static web page is requested, the web server can send the web page's content, without modification, to the requesting browser.

This simple model does not work for dynamic web pages, where the content of the page may depend on various factors that can differ on a per-visitor basis. To accommodate dynamic content, dynamic web pages contain source code that is **executed** when the page is requested (see Figure 1.2). When the code is executed, it produces HTML markup as its result, which is then sent back to and displayed in the visitor's browser.

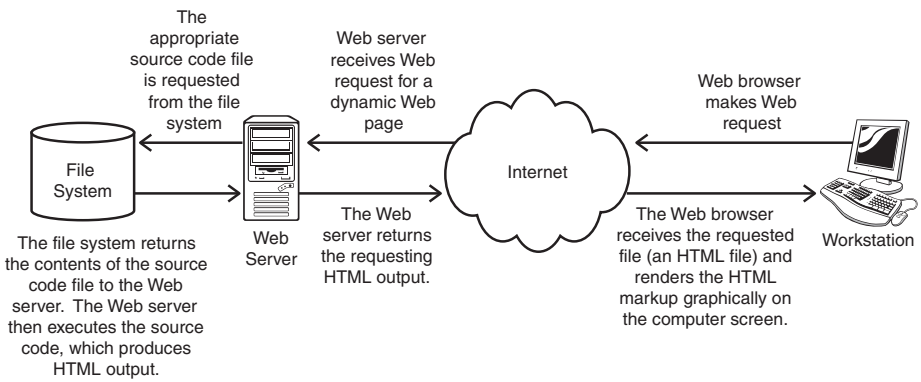

FIGURE 1.2
The content of a dynamic web page is created by executing the dynamic web page's source code.

This model allows for dynamic content because the content isn't actually created until the web page is requested. Imagine that we wanted to create a web page that displays the current date and time. To do this using a static web page, someone would need to edit the web page every second, continually updating the content so that it contained the current date and time. Clearly, this isn't feasible.

With a dynamic web page, however, the executed code can retrieve and display the current date and time. Suppose that one particular user visits this page on June 12, 2008, at 4:15:03 p.m. When the web request arrives, the dynamic web page's code is executed, which obtains the current date and time and returns it to the requesting

web browser. The visitor's browser displays the date and time the web page was executed: June 12, 2008, 4:15:03 p.m. If another visitor requests this page 7 seconds later, the dynamic web page's code will again be executed, returning June 12, 2008, 4:15:10 p.m.

Figure 1.2 is, in actuality, a slightly oversimplified model. Commonly, the web server and the execution of the dynamic web page source code are decoupled. When a web request arrives, the web server determines whether the requested page is a static web page or dynamic web page. If the requested web page is static, its contents are sent directly back to the browser that initiated the request (as shown in Figure 1.1). If, however, the requested web page is dynamic—for example, an ASP.NET web page—the web server hands off the responsibility of executing the page to the **ASP.NET engine** (see Figure 1.3).

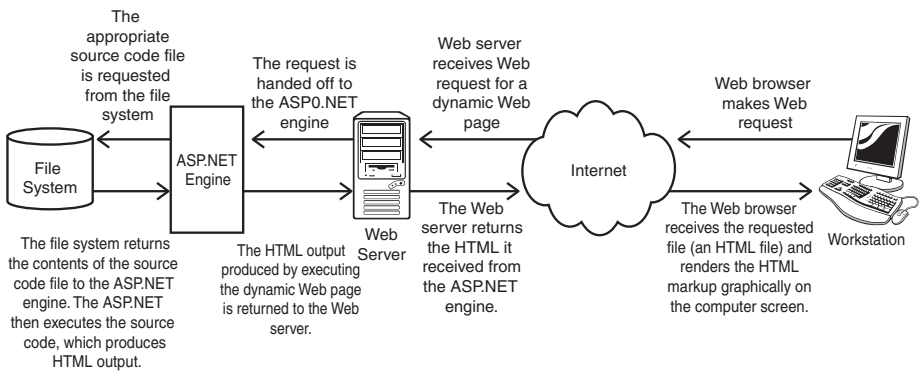

FIGURE 1.3 Execution of an ASP.NET web page is handled by the ASP.NET engine.

The web server can determine whether the requested page is a dynamic or static web page by the requested file's extension. If the extension is `.aspx`, the web server knows the requested page is an ASP.NET web page and therefore hands off the request to the ASP.NET engine.

By the Way

> The ASP.NET engine is a piece of software that knows how to execute ASP.NET web pages. Other web programming technologies, such as ASP, PHP, and JSP, have their own engines, which know how to execute ASP, PHP, and JSP pages.

When the ASP.NET engine executes an ASP.NET page, the engine generates the web page's resulting HTML output. This HTML output is then returned to the web server, which then returns it to the browser that initiated the web request.

Hosting ASP.NET Web Pages

To view an ASP.NET web page that resides on a web server, we need to request it through a browser. The browser sends the request to the web server, which then dispatches the request to the ASP.NET engine. The ASP.NET engine processes the requested page and returns the resulting HTML to the browser. When you're developing ASP.NET websites, the ASP.NET web pages you create are saved on your personal computer. For you to be able to test these pages, then, your computer must have a web server installed.

Fortunately, you do not need to concern yourself with installing a web server on your computer. Visual Web Developer, the editor we'll be using throughout this book to create our ASP.NET websites, includes a lightweight web server specifically designed for testing ASP.NET pages locally. As we will see in later hours, when testing an ASP.NET page, Visual Web Developer starts the **ASP.NET Development Web Server** and launches a browser that issues a request of the form: `http://localhost:`*`portNumber`*`/`*`ASP.NET_Page`*`.aspx`.

The `http://localhost` portion of the request tells the browser to send the request to your personal computer's web server, in contrast to some other web server on the Internet. The *`portNumber`* specifies a particular **port** through which the request is made. All web servers listen for incoming requests on a particular port. When the ASP.NET Development Web Server is started, it chooses an open port, which is reflected in the *`portNumber`* portion of the URL. Finally, the *`ASP.NET_Page`*`.aspx` portion is the filename of the ASP.NET page being tested.

Hosting ASP.NET pages locally through the ASP.NET Development Web Server has a number of advantages:

- **Testing can be done while offline**—Because the request from your browser is being directed to your own personal computer, you don't need to be connected to the Internet to test your ASP.NET pages.
- **It's fast**—Local requests are, naturally, much quicker than requests that must travel over the Internet.
- **Advanced debugging features are available**—By developing locally, you can use advanced debugging techniques, such as halting the execution of an ASP.NET page and stepping through its code line-by-line.
- **It's secure**—The ASP.NET Development Web Server allows only local connections. With this lightweight web server, you don't need to worry about hackers gaining access to your system through an open website.

The main disadvantage of hosting ASP.NET pages locally is that they can be viewed only from your computer. That is, a visitor on another computer cannot enter some URL into her browser's Address bar that will take her to the ASP.NET website you've created on your local computer. If you want to create an ASP.NET website that can be visited by anyone with an Internet connection, you should consider using a web-hosting company.

Web-hosting companies have a number of Internet-accessible computers on which individuals or companies can host their websites. These computers contain web servers that are accessible from any other computer on the Internet. The benefits of using a web-hosting company to host your site include

- **A publicly available website**—With a web-hosting company, any visitor who has an Internet connection can visit your website!
- **Use of a domain name**—You can register a domain name and have it point to your website so that visitors can reach your website through a name like www.mysite.com.
- **Ability to focus 100% on building your website**—Installing a web server, applying the latest security patches, properly configuring domain names, and so forth can be tricky tasks. By using a web-hosting company, you are paying for this service, which enables you to concentrate on building your website.

After you have settled on a web-hosting company and have set up your account, you are ready to move the ASP.NET pages from your computer to the web-hosting company. This process is referred to as **deployment**, and is covered in-depth in Hour 24, "Deploying Your Website." After a website has been successfully deployed, you, or anyone else on the Internet, can visit the site from their web browser.

Develop Locally, Deploy to a Web Host

Because there are different advantages for hosting a site locally versus hosting with a web-hosting company, often the best choice is to do both! I encourage you to develop, test, and debug your ASP.NET websites locally, through Visual Web Developer's built-in web server. After you have completed your site and are ready to go live, you can procure an account with a web-hosting company and deploy your site. This approach allows for the best of both worlds—an ideal development environment with the end result of a publicly accessible website!

The examples and lessons presented in Hour 1 through Hour 23 are meant to be created, tested, and debugged locally. Hour 24 contains step-by-step instructions for deploying your ASP.NET website to a web-hosting company.

Installing the ASP.NET Engine, Editor, and Database System

For a web server to be able to serve ASP.NET pages, it must have the ASP.NET engine installed. Recall that the ASP.NET engine is responsible for executing the ASP.NET web page and generating its resulting HTML. To install the ASP.NET engine, your computer must be running Windows XP, Windows Server 2003, Windows Vista, or Windows Server 2008. Even if your system does have the required operating system installed, you may need to take additional steps before you can start working with ASP.NET. For example, those using Windows XP need to have Service Pack 2 (SP2) installed. If you're uncertain whether your system meets the requirements, attempt the installation process. The installation program will inform you if there is some prerequisite for installation, such as Service Pack 2.

Three components need to be installed for us to work with the ASP.NET examples throughout this book. First, we must install the .NET Framework, which contains the core libraries required to execute an ASP.NET page. The ASP.NET engine is part of this .NET Framework. Following that, we need to install Visual Web Developer, which is the editor of choice for working with ASP.NET pages. Finally, we need to install SQL Server 2005, a database engine that is used extensively from Hour 14, "Accessing Data with the Data Source Web Controls," onward.

All three components can be installed through the single installation program included on this book's accompanying CD. To begin the installation process, insert the CD into your computer. This brings up the installation program starting with the screen shown in Figure 1.4. Click the Next button to progress through the subsequent two screens.

By the Way

If your computer lacks the prerequisites, such as not having the latest service pack installed, the installation program informs you of the problem and gives you instructions on how to update your system. After you have updated your computer, rerun the installation program.

The book's CD is the installation CD for Microsoft's Visual Web Developer editor. Because Visual Web Developer is designed for developing ASP.NET websites, the installation process automatically installs the .NET Framework and other required ASP.NET tools. You can also install three optional packages, as shown in Figure 1.5.

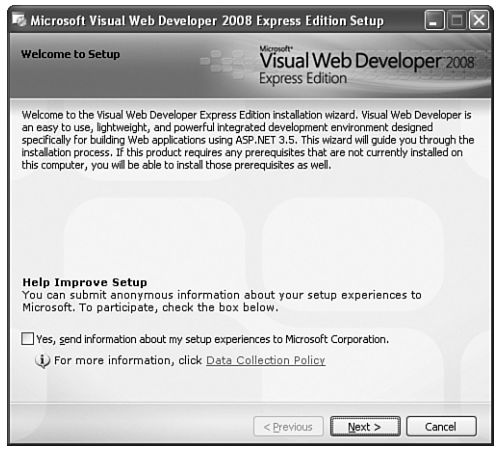

FIGURE 1.4 Start the installation process by inserting the CD into your computer's CD-ROM drive.

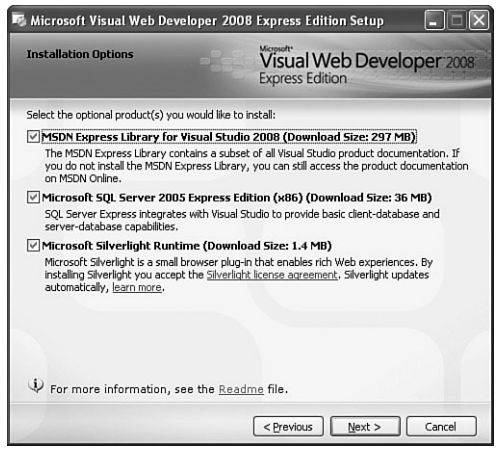

FIGURE 1.5 Make sure that you install SQL Server 2005 Express Edition.

The first optional package is MSDN Express Library for Visual Studio 2008. MSDN is Microsoft's collection of product documentation, whitepapers, code samples, and help files. Although all this information is accessible online at http://msdn2.microsoft.com, I encourage you to install MSDN locally on your computer. The local documentation can be searched and accessed quicker than its online counterpart and is still available even if your Internet connection is down.

The second optional package is Microsoft SQL Server 2005 Express Edition. This package is optional in the sense that Visual Web Developer will install successfully

with or without SQL Server 2005; however, the latter half of this book's examples rely on SQL Server 2005 being installed. Therefore, make sure that this check box is selected.

The third package is for Microsoft's Silverlight runtime. Silverlight is a browser plug-in from Microsoft that is capable of displaying rich, interactive multimedia content. The Microsoft Silverlight runtime is required to view Silverlight applications in your browser. None of the examples in this book use Silverlight, so you may choose to not install it.

By the Way

The Microsoft SQL Server 2005 Express Edition and Microsoft Silverlight Runtime check boxes will not be displayed if you already have these components installed on your computer.

The next screen (see Figure 1.6) enables you to specify in what folder to install Visual Web Developer as well as what products will be installed and the disk space required. After double-checking that the correct packages are being installed, click the Install button to begin the installation process. The overall installation process will take several minutes. During the installation, you are kept abreast with what package is currently being installed as well as the overall installation progress (see Figure 1.7).

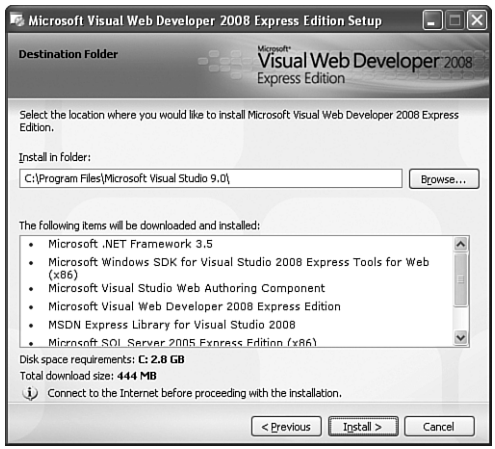

FIGURE 1.6 Specify the folder in which to install Visual Web Developer.

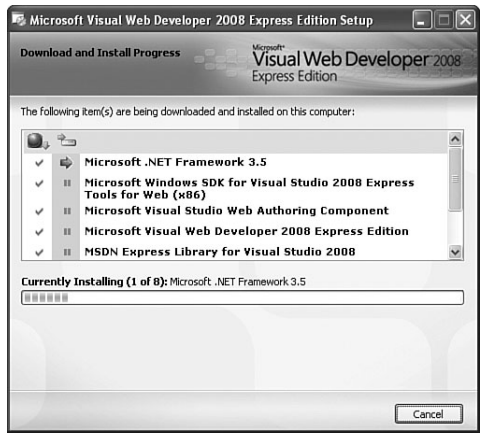

FIGURE 1.7 Monitor the installation's progress.

A Brief Tour of Visual Web Developer

When the installation process completes, take a moment to poke through Visual Web Developer. To launch Visual Web Developer, go to the Start menu, choose Programs, and click Microsoft Visual Web Developer 2008 Express Edition. Figure 1.8 shows Visual Web Developer when it loads.

When you open Visual Web Developer, the Start Page is initially shown. This Start Page includes a list of Recent Projects in the upper-left corner, a Getting Started section with some links for accomplishing common tasks in the bottom-left corner, and a list of recent articles on Microsoft's MSDN site in the right column.

On the left you'll find the Toolbox. When you view the Start Page, the Toolbox is empty, but when you work with an ASP.NET page, it contains the plethora of ASP.NET Web controls that can be added to the page. (We'll discuss what Web controls are and their purpose in the next hour.) Two other windows share the left region with the Toolbox: CSS Properties and Manage Styles. These windows are used to define style and appearance settings for the HTML and Web control elements within a web page.

To the right of the screen, you'll find the Solution Explorer. Again, on the Start Page this is empty, but when you load or create an ASP.NET website, the Solution Explorer will list the website's files. These files include database files, HTML pages, ASP.NET pages, image files, CSS files, configuration files, and so on. In addition to the Solution Explorer, the right portion of the screen is also home to the Database Explorer and Properties windows. The Database Explorer lists the databases associated with the project and provides functionality for creating, editing, and deleting the structure and contents of these databases. When you design an ASP.NET page, the Properties window displays information about the currently selected Web control or HTML element.

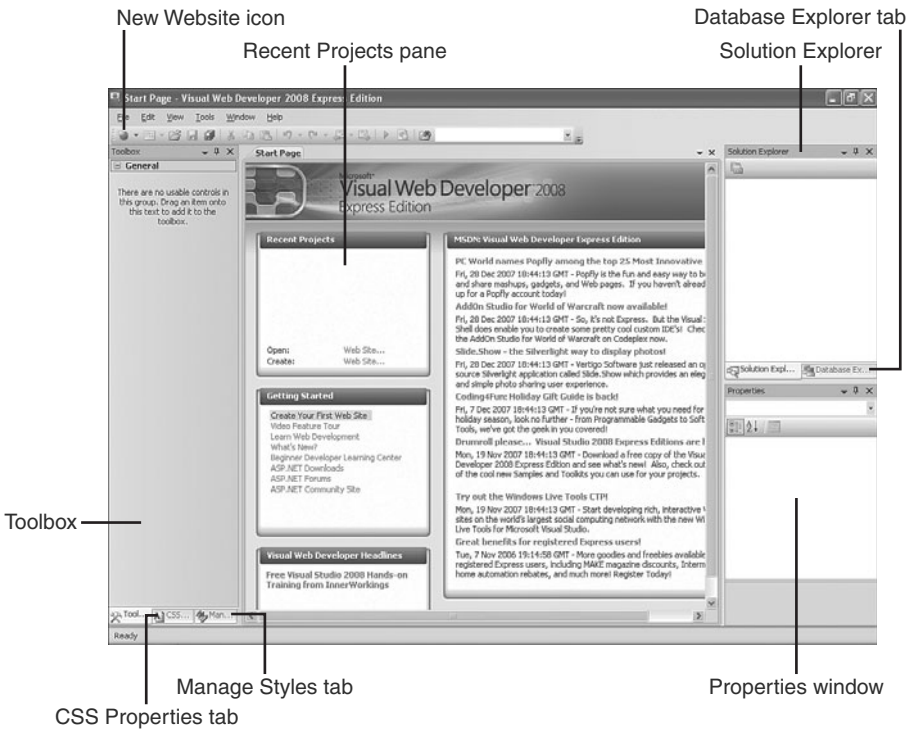

FIGURE 1.8 The Start Page shows when Visual Web Developer is loaded.

Creating a New ASP.NET Website

To create and design an ASP.NET page, we must first create an ASP.NET website. From Visual Web Developer, you can choose several ways to create a new ASP.NET website. You can go to the File menu and choose the New Web Site option; you can click the New Website icon in the Toolbar; or you can click the Create Web Site link in the Recent Projects pane of the Start Page.

All these approaches bring up the New Web Site dialog box, as shown in Figure 1.9. Let's take a moment to create a website. For now, don't worry about all the options available or what they mean because we'll discuss them in detail in Hour 3, "Using Visual Web Developer." Leave the Templates selection as ASP.NET Web Site, the Location drop-down list as File System, and the Language drop-down list as Visual Basic. The only thing you should change is the actual location of the website. Place the website in a folder named `MyFirstWebsite` on your desktop.

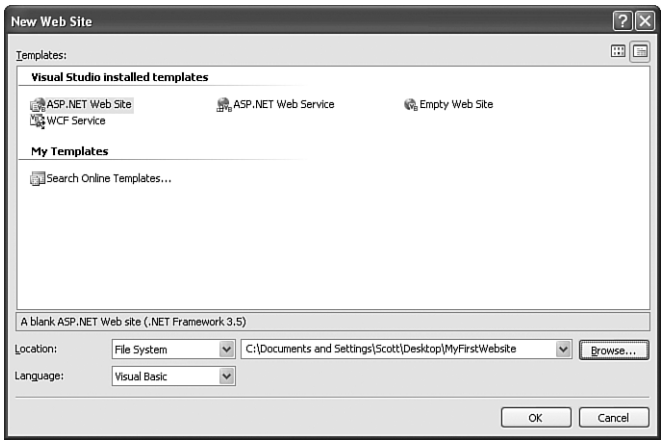

FIGURE 1.9 Create a new ASP.NET website in a folder on your desktop.

After you create the new website, your screen should look similar to Figure 1.10. When creating the new website, Visual Web Developer automatically created an `App_Data` folder, an ASP.NET page named `Default.aspx`, and a web configuration file (`web.config`). This folder and these two files are shown in the Solution Explorer.

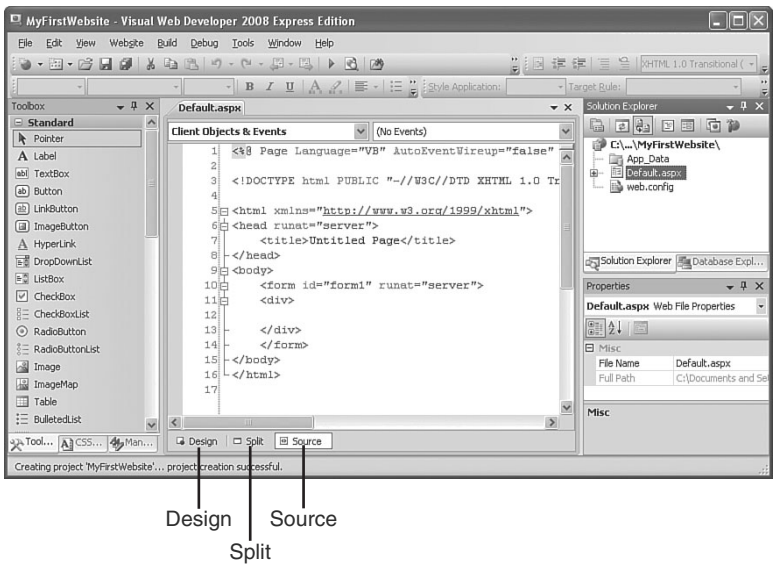

FIGURE 1.10 A new website has been created with an ASP.NET page, `Default.aspx`.

The `Default.aspx` page that was automatically created is opened and its contents are shown in the main window. Right now this ASP.NET page consists of just HTML. As we will see in future hours, ASP.NET pages can also contain Web controls and

server-side source code. Typically, ASP.NET pages are broken into two files: one that contains the HTML markup and Web control syntax, and another that contains just the code. In fact, if you click the plus icon on `Default.aspx` in the Solution Explorer, you'll see that there's another file, `Default.aspx.vb`, that is nested. This is the source code file for `Default.aspx`.

By the Way

Don't worry if you're feeling a bit overwhelmed. The point of this hour is to give a cursory overview of Visual Web Developer. Over the next three hours, we'll look at the portions of an ASP.NET page and the steps involved in creating and testing ASP.NET pages in much greater detail.

When you're working with the HTML elements and Web controls in an ASP.NET web page, you'll notice three views. The first is the Source view, which shows the page's underlying HTML markup and Web control syntax. This is the default view, and the one shown in Figure 1.10. The second view, called the Design view, provides a simpler alternative to specifying and viewing the page's content. In the Design view you can drag and drop HTML elements and Web controls from the Toolbox onto the design surface. You don't need to type in the specific HTML or Web control syntax. The third view, Split, divides the screen in half, showing the Source view in the top portion and the corresponding Design view in the bottom. You can toggle between the Source, Design, and Split views for an ASP.NET page by using the Design, Source, and Split buttons at the bottom of the main window.

Did you Know?

You can reposition the Properties, Toolbox, Solution Explorer, Database Explorer, and other windows by clicking their title bars and dragging them elsewhere, or resize them by clicking their borders. If you accidentally close one of these windows by clicking the X in the title bar, you can redisplay it from the View menu.

Creating and Testing a Simple ASP.NET Web Page

To view or test an ASP.NET web page, a browser needs to make a request to the web server for that web page. Let's test `Default.aspx`. Before we do, though, let's add some content to the page, because right now the page's HTML will not display anything when viewed through a browser. From the Source view, place your cursor between the `<div>` and `</div>` tags in `Default.aspx` and add the following text:

```
<h1>Hello, World!</h1>
```

This displays the text **Hello, World!** in a large font. After entering this text, go to the Debug menu and choose the Start Without Debugging menu option. This starts the

ASP.NET Development Web Server and launches your computer's default browser, directing it to `http://localhost:portNumber/MyFirstWebsite/Default.aspx` (see Figure 1.11). The *portNumber* portion in the URL will depend on the port selected by the ASP.NET Development Web Server.

FIGURE 1.11 `Default.aspx`, when viewed through a browser.

This ASP.NET page isn't very interesting because its content is static. It does, however, illustrate that to view the contents of an ASP.NET page, you must start the ASP.NET Development Web Server and request the page through a browser.

By the Way

You may be wondering what debugging is, and why I instructed you to start without it. We'll cover the differences between the Start Debugging and Start Without Debugging menu options in Hour 4.

Summary

Today virtually all websites contain dynamic web pages of one kind or another. Any website that allows a user to search its content, order products, or customize the site's content is dynamic in nature. A number of competing technologies exist for creating dynamic pages—one of the best is ASP.NET.

Throughout this book we'll be examining how to create interactive and interesting ASP.NET web pages. You will implement the examples using Visual Web Developer, a free editor from Microsoft designed specifically for working on ASP.NET websites. In the "Installing the ASP.NET Engine, Editor, and Database System" section, we looked at how to install Visual Web Developer, along with the .NET Framework and SQL Server 2005. In the "A Brief Tour of Visual Web Developer" section, we poked around Visual Web Developer and created our first ASP.NET web page, testing it

through a browser. In Hour 3 we'll explore the Visual Web Developer environment in much greater detail.

We have just begun our foray into the world of ASP.NET. Over the next 23 hours we'll explore the ins and outs of this exciting technology!

Q&A

Q. ***What is the main difference between a static and a dynamic web page?***

A. A static web page has content that remains unchanged between web requests, whereas a dynamic web page's content is generated each time the page is requested. The dynamic content is typically generated from user input, database data, or some combination of the two.

For an example of a dynamic web page, consider the official website for the National Basketball Association (www.NBA.com), which lists team schedules, ongoing game scores, player statistics, and so on. This site's web pages display information stored in a database. Another example is a search engine site such as Google, which displays dynamic content based on both its catalog of indexed websites and the visitor's search terms.

Q. ***You mentioned that with Visual Web Developer we'll be using the ASP.NET Development Web Server. Are other web server systems available for serving ASP.NET pages?***

A. The ASP.NET Development Web Server is designed specifically for testing ASP.NET web pages locally. The computers at web-hosting companies use different web server software. Many web-hosting companies use Microsoft's Internet Information Server (IIS), which is a professional-grade web server designed to work with Microsoft's dynamic web technologies—ASP and ASP.NET.

If you are running Windows XP Professional, Windows Server 2003, Windows Vista, or Windows Server 2008, IIS may already be installed on your computer. If not, you can install it by going to Start, Settings, Control Panel, Add or Remove Programs, and clicking the Add/Remove Windows Components option.

I encourage you not to install IIS and to instead use the ASP.NET Development Web Server unless you are already familiar with IIS and know how to administer and secure it. The ASP.NET Development Web Server is more secure because it allows only incoming web requests from the local computer; IIS, on the other hand, is a full-blown web server and, unless properly patched and administered, can be an attack vector for malicious hackers.

Workshop

Quiz

1. What is the difference between a static web page and a dynamic web page?
2. What is the purpose of the ASP.NET engine?
3. True or False: ASP.NET web pages can be served from computers using the Windows ME operating system.
4. What software packages must be installed to serve ASP.NET web pages from a computer?
5. When should you consider using a web-hosting company to host your ASP.NET web pages?

Answers

1. The HTML markup for a static web page remains constant until a developer actually modifies the page's HTML. The HTML for a dynamic web page, on the other hand, is produced every time the web page is requested.
2. When the web server receives a request for an ASP.NET web page, it hands it off to the ASP.NET engine, which then executes the requested ASP.NET page and returns the rendered HTML to the web server. The ASP.NET engine allows for the HTML of an ASP.NET web page to be dynamically generated for each request.
3. False. ASP.NET pages can be served only from computers running Windows XP, Windows Server 2003, or Windows Vista.
4. For a computer to serve ASP.NET web pages, the .NET Framework and a web server that supports ASP.NET must be installed.
5. You should consider using a web-hosting company if you want your ASP.NET website to be accessible via the Internet. Typically, it's best to first develop the website locally and then deploy it to a web-hosting company when you are ready to go live.

Exercises

This hour does not have any exercises. We'll start with exercises in future hours, after we become more fluent with creating ASP.NET web pages.

Index

Symbols

A

B

C

D

E

F

G

H

I

J-K

L

M

N

O

P

Q

R

S

T

U

V

W-X-Y-Z