1 Getting Started with WebMatrix

WHAT YOU WILL LEARN IN THIS CHAPTER:

- What WebMatrix is all about
- How to acquire and install WebMatrix
- How to create a simple site with WebMatrix
- Where to go within WebMatrix to get things done
- What file types you are likely to be working with
- How to structure a website

In the early days of web development on the Windows platform — in the 90s of the last decade — Microsoft offered a relatively simple and approachable technology, now referred to as Classic ASP, which enabled inexperienced web developers and even non-programmers to build simple websites. Getting started with Classic ASP was easy; all you needed was a text editor and a hosting account to run your site. With the release of ASP.NET and Visual Studio .NET in early 2002, the web development landscape changed considerably. Although extremely powerful, ASP.NET is not easily approachable, and certainly not so for non-programmers. It has a pretty steep learning curve and requires experience in programming. The tool used to build ASP.NET websites — Visual Studio — is also a lot more complex to use than a simple text editor. In addition, it takes a fair bit of time to download and install Visual Studio.

Although the advent of ASP.NET and Visual Studio addressed the needs of experienced and professional programmers and was generally seen as a major leap forward, Microsoft no longer had a good option for people starting out with web development. To accommodate this group of users and make it easier for them to start developing websites on the Windows platform, Microsoft developed *WebMatrix* that was first released in January 2011. In this book, you'll get an in-depth look at WebMatrix and how to use it, starting with instructions on acquiring and installing WebMatrix in this chapter, all the way down to deploying a fully functional website to a production environment in Chapter 14.

In the next section you'll learn what WebMatrix is and how to install it. The section that follows introduces you to ASP.NET Web Pages — the development framework you'll be using in WebMatrix. The second half of this chapter then gives you an extensive tour of the WebMatrix user interface.

INTRODUCING WEBMATRIX

In this introductory section you'll learn what WebMatrix is and why you should use it. In addition, you'll learn how to acquire and install it, setting you up for the many exercises you'll find throughout this book.

What Is WebMatrix?

Many people tend to refer to WebMatrix as the lightweight tool to create web pages using the new Web Pages Framework and Razor syntax. However, WebMatrix is more than just the tool. WebMatrix is a stack of software components required to build web applications delivered seamlessly in one package. There are four core components to the stack: a web server (IIS Express); a development framework (.NET 4.0); a database platform (SQL Server Compact Edition 4.0); and a lightweight web authoring and management tool.

IIS Express is a lightweight web server used to serve up pages and other requests made by the browser. It offers all of the core features of the full version of IIS 7, the web server that is typically used in production hosting scenarios on server editions of Windows. It doesn't require administrator privileges to run, nor does it require complex configuration. It can run on any operating system from Windows XP upwards (including Home Edition).

The .NET Framework is a huge collection of code libraries, which offer pre-built solutions to many, many common programming problems. The framework also includes a common language infrastructure, which enables developers to pick from many languages to write their applications in. The most popular languages are C# and VB. ASP.NET is a web development framework that sits on top of the .NET Framework.

SQL Server Compact Edition (SQL CE) is a file-based database system, which you use to store your data in. Since it is file-based, SQL CE does not require you to run a setup program or install and configure a database server in order to use it. Databases are stored as files on disk with an SDF extension (similar to an Access database), and are easily copied to a web server or other machine via FTP, X-Copy, or similar. The database engine runs in memory within the application and shuts down automatically when it is no longer needed. It has a relatively slim set of features compared to the full version of SQL Server, which makes it very easy to work with.

The WebMatrix UI provides a host of tools for creating websites from templates or from scratch, downloading pre-built Open Source applications and customizing them, managing files, databases, publishing your web application to a web server, testing and configuring your site, optimizing your site for search engines, and more.

Why Should You Use WebMatrix?

There is a wide choice of programming languages and frameworks available to those who are just beginning web development, including PHP, Ruby on Rails, Java, ColdFusion, and even classic ASP — so why choose WebMatrix?

WebMatrix provides a relatively simple approach to web development, which can help you become productive very quickly. But don't let the simplicity of the development tool fool you. Behind the scenes you have access to the full power of the .NET Framework. The .NET Framework is composed literally of thousands of libraries of pre-written code covering nearly every programming requirement you will ever need. Microsoft is constantly expanding the framework, and of course fully supports it. Furthermore, a huge and active community of volunteers actively supports ASP.NET at places such as Wrox's own forums (`http://p2p.wrox.com/`) and the official Microsoft ASP.NET forums (`http://forums.asp.net`), where the authors of this book actively participate and try to answer your questions.

When planning WebMatrix, the development team at Microsoft decided that simplicity is key, and everything they do is driven by a desire to keep the "concept count" low for beginners to web development. There is stuff that every developer of dynamic websites needs to know, regardless of the framework they choose: HTML, CSS, some JavaScript, SQL, and server-side code. To varying degrees, all frameworks obscure some of the unnecessary details relating to these fundamentals. In doing so, some frameworks introduce a new range of concepts to learn and become familiar with. The upside of this is that once the new concepts are learned, you can enjoy a consistent development experience that encourages RAD, or Rapid Application Development. The downside is that learning the new concepts can take a long time and might overwhelm the beginner.

WebMatrix goes back to the roots of web development. It encourages you to immerse yourself in HTML, CSS, JavaScript, and more, which are accessible technologies common to all web development platforms. It also includes a range of "helpers," which are wrappers around some of the common tasks you will perform when developing a website, such as data access, managing security, and sending e-mail. These helpers provide shortcuts more than anything, and they are very easy to learn and use. You typically access these helpers from your web pages using either Visual Basic or C# code. The examples in this book all use C# as the language of choice, but you could follow along with Visual Basic if you prefer.

Another great feature of WebMatrix is its price: It's completely free. This means you can follow along with all the examples from this book, and build great production-ready websites without spending a dime.

How Do You Acquire WebMatrix?

In theory, it's possible to create ASP.NET Web Pages applications using any text-editing package, such as Notepad. In theory, it's also possible for you to commute to work or college on your hands and knees. I wouldn't recommend either as a sensible option. Get a bus or train, drive a car, or cycle to work or college, and use WebMatrix for Web Pages development.

Usually, before you attempt to download and install all the bits that form a web development framework, you would be advised to check that your system meets the minimum requirements. Then you might find yourself hopping around the Internet locating and downloading the various parts that make up the whole. But those days are gone. Microsoft released a product called the Web Platform Installer (WPI), which simplifies the whole process. WPI analyzes your system and identifies if parts of what you need are already installed. It knows which items are dependent on which other items, so all you need to do is select WebMatrix from the list of options, and let WPI do all the work. Your development machine's operating system must be compatible with version 4.0 of the .NET Framework which means you can use any of the following operating systems:

- Windows 7
- Windows Server 2008
- Windows Server 2008 R2
- Windows Vista SP1 or later
- Windows XP SP3
- Windows Server 2003 SP2

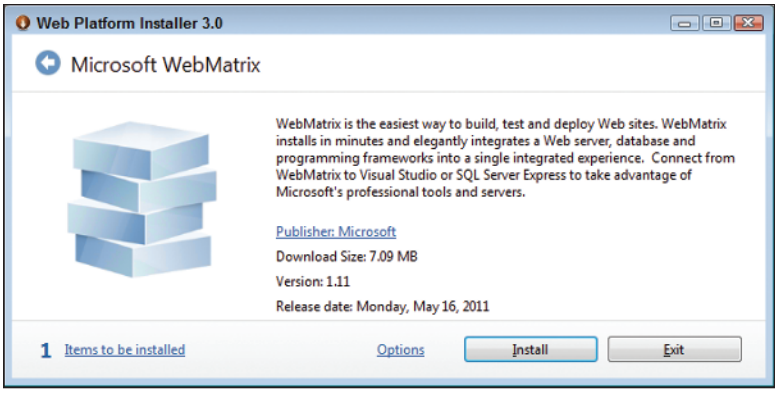
NOTE *When you install WebMatrix, you also get a copy of IIS Express, the lightweight web server. This web server requires your Documents folder to be on a local hard drive. You'll run into problems if this folder is stored on the network. If this is the case, locate your Documents folder in Windows, right-click it and choose Properties and switch to the Location tab. You can click the Move icon to move your Documents folder to a local drive, such as your C drive. You can choose to move your existing files to this new location, but this is not required.*

WebMatrix is a small download — about 15MB if you already have .NET 4.0 installed, rising to about 50MB if WPI needs to fetch .NET 4.0 for you as well.

TRY IT OUT Installing WebMatrix

This exercise guides you through the simple process of acquiring, downloading, and installing WebMatrix using the Web Platform Installer.

1. You can download WebMatrix via the Web Platform Installer (WPI) from this Microsoft site: `www.microsoft.com/web/webmatrix/`. If this link ever changes, you can also find the WPI at `www.microsoft.com/web`. If none of these links works, search the main `Microsoft.com` website for "WebMatrix" or "Web Platform Installer."
2. At the WebMatrix or WPI website, look for a button such as Install WebMatrix or Install Now and click it. This starts the WPI, which guides you through installing WebMatrix. If you started your download at the WebMatrix site, you'll see a screen similar to the one shown in Figure 1-1.

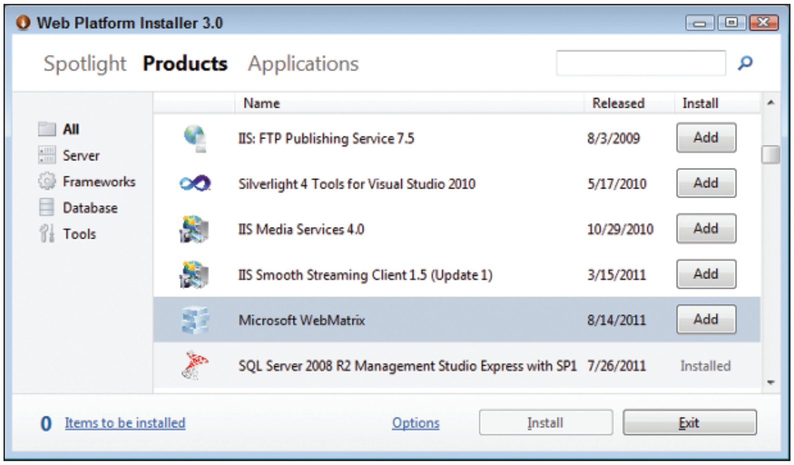


WebMatrix is the easiest way to build, test and deploy Web sites. WebMatrix installs in minutes and elegantly integrates a Web server, database and programming frameworks into a single integrated experience. Connect from WebMatrix to Visual Studio or SQL Server Express to take advantage of Microsoft's professional tools and servers.

Publisher: Microsoft
Download Size: 7.09 MB
Version: 1.11
Release date: Monday, May 16, 2011

FIGURE 1-1

3. Click Install and follow the on-screen instructions to complete the install. If you started a WPI download instead, you'll see a screen similar to Figure 1-2.



Name	Released	Install
IIS: FTP Publishing Service 7.5	8/3/2009	Add
Silverlight 4 Tools for Visual Studio 2010	5/17/2010	Add
IIS Media Services 4.0	10/29/2010	Add
IIS Smooth Streaming Client 1.5 (Update 1)	3/15/2011	Add
Microsoft WebMatrix	8/14/2011	Add
SQL Server 2008 R2 Management Studio Express with SP1	7/26/2011	Installed

FIGURE 1-2

4. Locate Microsoft WebMatrix in the list (you can use the Search box at the top to find it), add it to the list of downloads by clicking Add, and then click the Install button.

   No matter how you started the installation of WebMatrix, your files are now downloaded and installed for you. Within minutes (depending on your Internet connection speed), WebMatrix should be completely installed, and you're ready to roll with the remainder of this book.

How It Works

Rather than going to many different websites finding the latest bits and pieces you need to install separately, Microsoft has released the Web Platform Installer, your one-stop download and installation tool for many of Microsoft's developer applications, such as WebMatrix and Visual Web Developer Express

Edition, as well as many Open Source tools and frameworks. The first time you install Web Platform Installer, it also installs an icon in your Start menu, making WPI readily available in case you need to install additional software. Besides WPI, you'll also see items for WebMatrix in your Start menu. You won't see any additional icons for the other installation dependencies, such as IIS Express or SQL Server CE. You'll learn how to access these tools from within WebMatrix later in this book.

Introducing the ASP.NET Web Pages Framework

ASP.NET Web Pages is a web development framework. It is built on top of the existing ASP.NET Framework, which in turn is built on top of the .NET Framework. The ASP.NET Framework provides an extraordinarily powerful development experience for web developers. Common tasks are wrapped into easy-to-use classes. Authentication, membership and roles management, file management, data access… all of these have their own APIs (Application Programming Interfaces), which make working with them very easy compared to other approaches. The ASP.NET Framework was conceived with the enterprise developer in mind, and reflects the demands that maintaining large, ever-changing sites impose.

Traditional ASP.NET development encourages a separation of code from the presentation layer, using a model called "code behind." Code is placed in separate files in the same way developers of desktop applications had been doing for years. User Interface (UI) elements result from the rendering of so-called server controls, which emit HTML to the browser, shielding the developer from building up that HTML manually.

Other web development frameworks follow a different pattern, in that the HTML is not hidden away. You work directly with it. The Web Pages Framework follows this pattern, which means that the development experience is not unlike the one that PHP or classic ASP developers are used to. Server code is intermixed with HTML using the Razor syntax, which again is a similar experience to other frameworks. And it is easier to learn, mainly because there is not so much to absorb. Razor syntax is a very succinct way to define markup (such as HTML) mixed with server side programming logic. Chapter 4 gives you an in-depth look at Razor and how to use it in your WebMatrix websites.

Despite the fact that Web Pages is easier to learn, it still sits on top of the .NET Framework (and the ASP.NET Framework), which means that sites you develop using Web Pages can be as complex and functional as any built by enterprise developers. The key point, however, is that Web Pages makes building simple sites a simple task. It removes a lot of the configuration steps involved in building sites using either the Web Forms model or the newer MVC model. You simply open WebMatrix, create a new site, and start coding — as you'll see how to do in the next exercise.

TRY IT OUT Creating Your First Web Pages Application

Now that you have downloaded and installed WebMatrix, it's a good time to build a simple web application to test that everything works correctly on your system. This section shows you how to do that, and then explores a bit of the details behind it in the "How It Works" section.

1. Start WebMatrix from the Windows Start menu. You should be presented with the Quick Start screen, as shown in Figure 1-3.

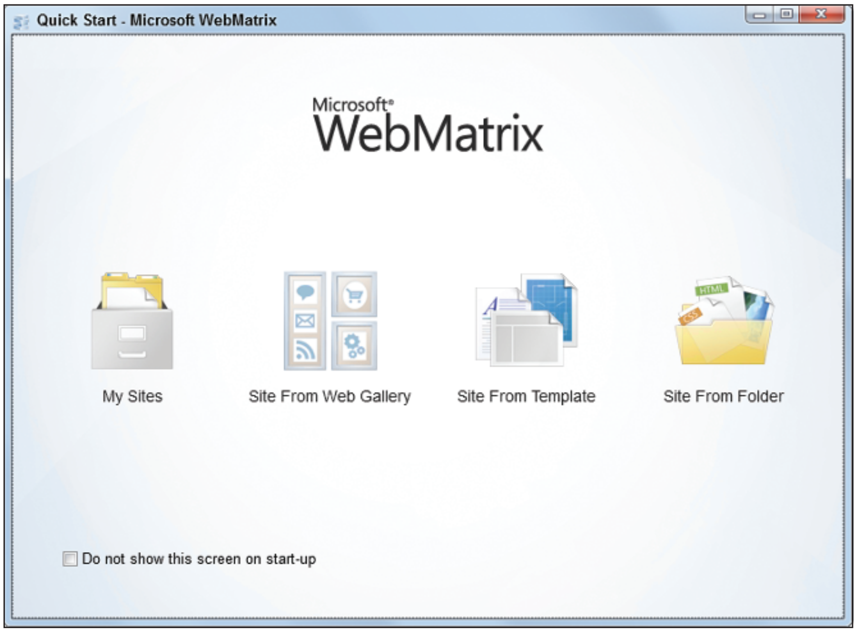


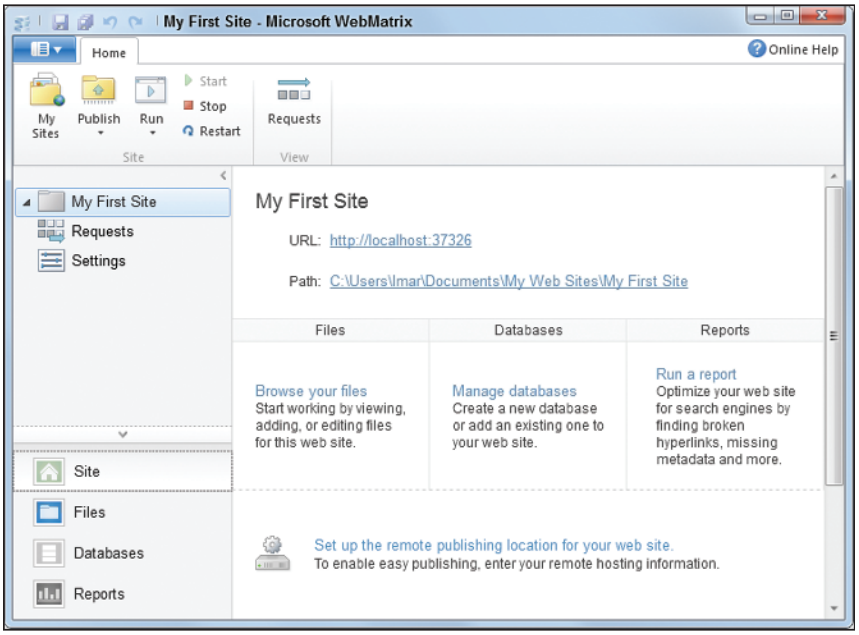
FIGURE 1-3

Notice the option in the bottom-left corner where you can choose not to see this screen in the future. If you accidentally turned it off now, you can turn it on later by clicking the main WebMatrix button and then choosing Options. You find the option to turn on the Quick Start screen in the General Options for Working with WebMatrix section.

2. Ignoring the other three options for the moment, click Site From Template.
3. Click the Empty Site template to select it from the options available. You can name this site anything you like in the Site Name box at the bottom. For this exercise, call the site **My First Site**, and click OK. You should see a message that your site is being created from the chosen template, which is then replaced with a confirmation that your site has been started successfully. The confirmation message appears temporarily at the bottom of the screen.
4. By default, WebMatrix always opens your site with the Site option selected in the left pane and the Site workspace visible, as shown in Figure 1-4.

   You will explore the various workspaces later in the chapter, but for now, click the Files button in the lower-left corner. You will be shown a screen that invites you to add a new file to your site. You can also see that the Empty Site template is an appropriate name — there are no files there to begin with except a file called robots.txt, which is explained later in this book.
5. There are three ways to add a new file to your site. The first is simply to click the link in the middle of the workspace. The second option is available from what is called the *Ribbon Bar* — the main toolbar at the top of the WebMatrix window. This hosts a collection of icons and text used to represent many of the tasks and actions you can perform with WebMatrix. In the second group of items — Files — you can see the word New under a document icon. If you click that, you are offered

the option to add a new file or folder. Finally, you can right-click on the site name at the top of the left pane. The first option provided by the context menu is New File. For now, though, just click the link in the middle of the workspace.



My First Site

URL: http://localhost:37326

Path: C:\Users\Imar\Documents\My Web Sites\My First Site

Browse your files
Start working by viewing, adding, or editing files for this web site.

Manage databases
Create a new database or add an existing one to your web site.

Run a report
Optimize your web site for search engines by finding broken hyperlinks, missing metadata and more.

Set up the remote publishing location for your web site.
To enable easy publishing, enter your remote hosting information.

FIGURE 1-4

6. The next step is to choose a file type. The options available to you will depend on which category — Common, Suggested, or All — is selected at the top left of the screen. You will explore the various file types later in this chapter. In the meantime, choose CSHTML, which is a *C# Razor* file; leave the filename to its default of Page.cshtml; and click OK.

7. Add the following bold text and code between the opening and closing `body` tags:

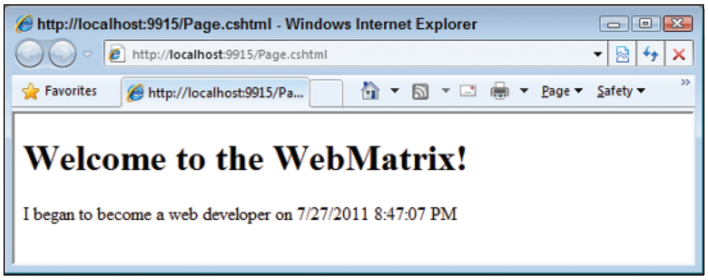
```
<body>
  <h1>Welcome to the WebMatrix!</h1>
  <p>I began to become a web developer on @DateTime.Now.ToString()</p>
</body>
```

   Whenever you see bold text and code in this book, you only need to type that. Any non-bolded text should already be present in the file you are working on, if you have followed along correctly.

   Don't let the `@` sign bother you too much at this point. This is an intrinsic part of the Razor syntax, and over the next few chapters, you are going to become great friends with it.

8. Press Ctrl+S to save the changes to the page, and press F12 to run it in your default browser. If all goes as planned, you should see something similar to Figure 1-5.

   If you get an error message instead, there are two possible causes. If the error summary message is `HTTP Error 403.14 - Forbidden` or `HTTP Error 404.20 - Not Found`, you have probably not requested your new page directly, but its parent folder. In that case, go back to WebMatrix, double-click your page in the Files list, and try again.


http://localhost:9915/Page.cshtml - Windows Internet Explorer
http://localhost:9915/Page.cshtml
Favorites
http://localhost:9915/Pa...
Page
Safety

Welcome to the WebMatrix!

I began to become a web developer on 7/27/2011 8:47:07 PM

FIGURE 1-5

If the error message relates to a compilation error, chances are that you have a spelling mistake in your code. C# is case-sensitive, so make sure that you have used exactly the right combination of upper- and lowercase letters in `DateTime.Now.ToString()`.

NOTE *If you get a message from Internet Explorer about Intranet settings, click the button that lets you turn on these Intranet settings. You may want to follow the Help option first to learn more about this feature.*

9. Assuming that you managed to run the page successfully, make a note of the time that you began your web development career, and then refresh the page in the browser by pressing F5. You should see that the time has just been updated.

Congratulations, you just completed your first web application!

How It Works

When you added the CSHTML file to the site, it started life with some *HTML* or *Hypertext Markup Language* already added by WebMatrix by default. The two lines of code that you added consist of a mixture of more HTML (the items enclosed in angle brackets), static content ("Welcome to the WebMatrix" and "I began to become a web developer on") and some server-side code — the part preceded by the `@` symbol. When you requested the page in your browser, the web server — IIS Express — was called into action. Its job is to serve web pages as HTML so that the browser can understand the result and display it. HTML is a language that is used mainly by web browsers like Internet Explorer and Firefox so that they can recognize the structure and content of a document. IIS Express comes preconfigured to recognize the file types that are being requested and served. Some file types, such as images and static HTML files (ending with HTM or HTML), are served without any fuss at all. Others, like C# Razor files (with a CSHTML extension), are mapped within IIS Express to separate processing engines to transform the file from its raw server-side format (with the code blocks in it) to something that the browser understands. In the case of CSHTML files, they are mapped to ASP.NET. On your development box, the client (your browser) and the server (IIS Express) are located on the same physical machine. After you've put your site in production, the client and server parts of your site will be on different machines. Your site's files are hosted and served up on a server running IIS (the web server on Windows machines) while clients (potentially from all over the world) make requests for your pages using their browser. Chapter 14 digs deeper into deployment and shows you how to copy your site to one of the on-line services offering WebMatrix hosting facilities.

ASP.NET's role is to examine the content of the file and identify whether any server-side code embedded within it needs to be executed. You will most often use server-side code to generate content dynamically, depending on certain conditions. In the previous "Try It Out," the dynamic content was the current date and time. When you refreshed the page, the content was updated according to the system clock on the server at the point the code was executed.

If you have closed the browser, press F12 again or click the Run button on the Ribbon Bar in WebMatrix.

Now right-click anywhere on the page in the browser and select the relevant browser menu option to view the page source. The HTML source code that reached the browser should open in your default text editor. If you examine it, you can see that nearly all of it is identical to the code in the CSHTML file you created, except that `@DateTime.Now.ToString()` has been replaced with the current date and time in plaintext.

In the following section you'll get a more detailed look at what happens when you request a page in your browser.

How the Web Works

One of the most important aspects to becoming a successful web developer is to understand the relationship between the user's web browser and the web server. Judging by the type of questions that regularly appear in the ASP.NET forums — even from people who seem to have been programming websites for some time — this relationship is very often misunderstood or not considered at all. Although you saw some aspects of this relationship in the preceding section, it's now time to examine this issue in more detail.

Imagine that you are sitting in a restaurant. The waiter comes over and asks you for your food order. He takes out his pad and notes which table you are sitting at, and details of the menu items you would like. Then he disappears. A while later the waiter reappears with your order. You don't know where your meal came from or how it was prepared. You probably don't care as long as the meal is edible.

While he or she was gone, the waiter passed your food order to the chef in the kitchen. The chef sprang into action and prepared the order. A part of the order may have been fulfilled without any real work, such as a plain bread roll. Or the order may have needed special processing, such as frying or grilling. These tasks would have been passed to specialists within the kitchen to manage. Once complete, the order is passed to the waiter for delivery. The chef doesn't know what happened to the food that was prepared or where it went. He doesn't care (as long as he receives no complaints). His job is purely to process incoming requests and serve appropriate responses.

Web applications are like this. The browser or client plays the diner's role. The chef is the web server, the waiter is the transport mechanism for the *request* (the meal order) and the *response* (the prepared meal), and the waiter's order pad is the equivalent of the *HyperText Transfer Protocol* (HTTP). The client makes a request, which is conveyed to the correct web server, given the URL of the request. The server examines the request and decides if it can be fulfilled without further ado, or whether some processing is required. The part of the request that is examined primarily is the file extension. This may have been registered with the server and mapped to special processors, such as ASP.NET or PHP; or, in the case of an image or HTML file, there is no need for processing. The raw content of the file is served as a response.

Web Pages files (CSHTML and VBHTML) are mapped to the *ASP.NET run time*. This is part of the Microsoft .NET Framework, which is designed specifically to process web requests. When the ASP.NET run time is asked to process a Web Page file, it looks for two things:

- **Static Content:** This includes HTML and textual content. It may also include embedded styling information contained in CSS and JavaScript code. These elements are sent to the browser directly — exactly as they appear in the original file.
- **Dynamic Content:** In the previous "Try It Out," you saw that you can embed programming logic inline within HTML in a Web Pages file. Although you used C# in the example, you could do the same with VB. As you progress through this book, you will discover that you can also place code in special blocks or in separate files altogether. ASP.NET is responsible for executing and processing this code, which will most likely produce different results depending on the conditions within the code. For example, you might use code to perform calculations, detect who the current user is, and see if she is permitted to view the current page, hide or reveal data, or process user input supplied via forms. You will see this in a lot more detail beginning in Chapter 4, "Programming Your Site." Before that, Chapter 3, "Designing Websites," shows you how this type of code can help you manage the appearance of your site consistently across pages.

Once the file has been processed and the finalized response is ready, this is sent to the browser for consumption. The web server has no further contact with the browser. It just sits there, waiting for the next request.

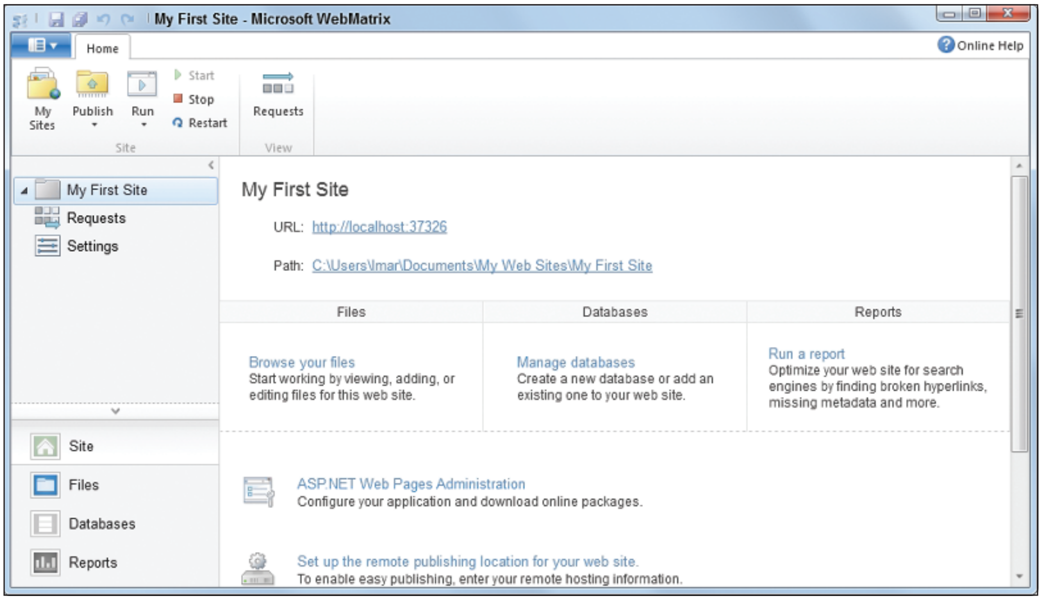
To get the most out of WebMatrix, you also need to be familiar with its user interface and the many buttons, panels, and workspaces it contains. In the next section, you'll take a look at the WebMatrix UI, which will prepare you to build your first real site in the following chapter.

A TOUR OF WEBMATRIX

WebMatrix is a very lightweight, but powerful, integrated development environment (IDE). It has been developed specifically for building ASP.NET Web Pages applications, but it can also be used for ASP.NET Web Forms, ASP.NET MVC, PHP, and classic ASP applications. It is known as an *integrated* development environment because it brings together all the tools you need to develop web applications in one place. Each of the tools has its own workspace, and each of the workspaces is accessible through the Workspace Selector at the bottom of the left pane. Before examining each workspace in detail, close WebMatrix if you haven't done so already, and then start it up via the Start menu in Windows. If you ticked the box to prevent the Quick Start screen from appearing in future, WebMatrix should open the last site you worked on. At the moment, this should be "My First Site." If you made no changes to the Quick Start screen, you should be presented with the same four options as illustrated in Figure 1-3, previously. This time choose My Sites, and then select My First Site and click OK. You will be taken to the first workspace — the Site workspace.

The Site Workspace

The primary purpose of the Site workspace is to provide you with a simplified one-page dashboard for accessing the key tasks and information related to managing your sites. To help you familiarize yourself with this workspace, take a look at Figure 1-6, which highlights the main features.



My First Site

URL: http://localhost:37326

Path: C:\Users\Imar\Documents\My Web Sites\My First Site

Files	Databases	Reports
Browse your files Start working by viewing, adding, or editing files for this web site.	Manage databases Create a new database or add an existing one to your web site.	Run a report Optimize your web site for search engines by finding broken hyperlinks, missing metadata and more.

ASP.NET Web Pages Administration
Configure your application and download online packages.

Set up the remote publishing location for your web site.
To enable easy publishing, enter your remote hosting information.

FIGURE 1-6

The Ribbon Bar

At the top of the screen, just below the Windows title bar, you'll see the familiar Ribbon Bar found in many other everyday Microsoft applications. The Ribbon Bar can be divided in tabs, and each tab can contain a number of groups. The only tab available in Figure 1-6 is the Home tab. The first group — Site — stays visible regardless of which workspace you are in, but the other options change depending on the task you perform. You will take a more detailed look at the functions available from the Site group later in the book, along with the purpose and features of the Requests option to the right of the Site group.

Left Pane

The left pane is collapsible. If you click the little arrowhead in the top-right corner, you can expand the right pane by collapsing the left one. Clicking the arrow again reverses this operation. The thin vertical line between the left and right panes is a splitter, and enables you to increase or decrease the width of the panes without collapsing the left one completely by dragging it to the left or right. In the Site workspace, the left pane contains two options — Requests and Settings. The Requests option is duplicated from the Ribbon Bar. Both items will be examined later in the book.

Workspace Selector

The Workspace Selector appears in the same position in all workspaces, below the left pane, and provides easy navigation between the four workspaces. This area is also collapsible.

Right Pane

This is the main work area within WebMatrix. When in the Site workspace, the right pane provides you with details of the currently selected site. These include its URL, which is typically `http://localhost` followed by a random port number. IIS chooses the port number. It is usually a large number, and one that outside sources find difficult to detect, for security reasons. If you click this link, your current site will fire up in a web browser.

You can also see the file path to the current site. By default, when you create a new site, it is added to a folder called My Web Sites within your Documents folder. Clicking this link will open the location of your current site within Windows Explorer. The three options that follow are alternative ways to navigate to the other three workspaces.

The final options in the bottom part of the workspace relate to configuring your site, obtaining a web hosting account if you don't already have one, and publishing your site to a live web server. You will make use of these options in Chapter 14, "Deploying your Site."

At the bottom of the Site workspace is the notifications area, which shows various messages such as whether your current site has been started by IIS or has been stopped. This area is not always visible, as the notifications come and go as appropriate.

If you press and hold down the Alt key, something rather interesting happens, as shown in Figure 1-7.

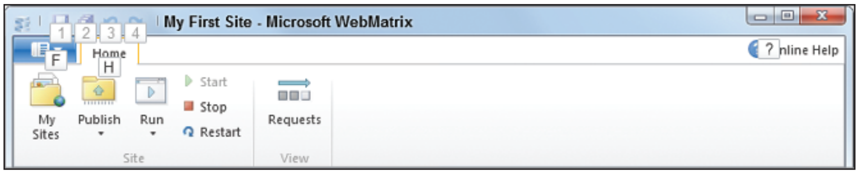
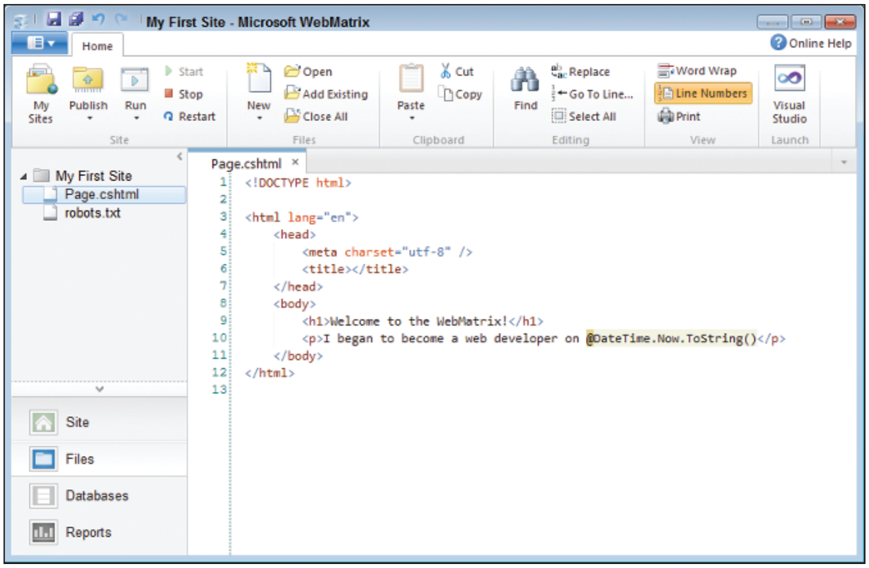


FIGURE 1-7

Just like in Microsoft Office and some other Microsoft applications, letters and numbers appear in specific positions on the Ribbon Bar. These are shortcut keys to the actions on the Ribbon Bar. For example, if you now press H, more letters appear on each item in the various Ribbon Bar groups, providing keyboard shortcuts to the options available. Note that this behavior is not limited to the Site workspace; you can use it anywhere in WebMatrix.

The Files Workspace

The Files workspace (shown in Figure 1-8) is where you are likely to spend most of your time within WebMatrix.



```
<!DOCTYPE html>

<html lang="en">
    <head>
        <meta charset="utf-8" />
        <title></title>
    </head>
    <body>
        <h1>Welcome to the WebMatrix!</h1>
        <p>I began to become a web developer on @DateTime.Now.ToString()</p>
    </body>
</html>
```

FIGURE 1-8

You can use the Files workspace to add new folders to your site, and work with files from the many types available, including Razor files, HTML files, CSS and JavaScript files, a site configuration file, VB or C# code files — even text and XML files. The options on the Ribbon Bar provide many common text-processing functions associated with working with files. The final option allows you to open the current site in Visual Web Developer 2010 Express Edition or Visual Studio 2010 if you have either of those tools installed. You will see how this could be useful in Chapter 9, "Debugging and Error Handling."

The left pane displays the entire folder structure of your website in a tree view. Files are opened in separate tabs within the right pane, providing you with the ability to have multiple documents opened at any one time. Each opened document displays its filename in the tab itself. You can navigate from one tab to another in a number of ways: You can simply click the tab you want; you can double-click the file in the tree menu in the left pane; you can click the down arrow at the top-right corner of the right pane and select a file from the menu that appears; or you can cycle through all the currently opened documents by pressing Ctrl+Tab.

The Databases Workspace

You will explore the databases workspace in a lot more detail in Chapter 10, "Introduction to Data and Databases." For the time being, all you really need to know is that this workspace makes working with databases very easy. You can create new databases from here and modify them by adding tables and columns. Once you have added database tables, you can open them and view, add, and edit data within them. You will be able to test your queries to ensure that they return the correct data prior to coding them into your website, and there is even a tool for migrating your SQL CE database to the full version of SQL Server.

The Reports Workspace

Once you have built your site, you will probably want people to visit it, but how are they going to know your site exists? The best way to publicize your site is to submit it to search engines. However, will the search engines like your site? Will they understand it? Will they be able to categorize your site in the correct search results? Search Engine Optimization is the practice of making websites highly accessible to search engines so that they index your site for the most appropriate search phrases. Chapter 13, "Optimizing Your Site," explores this topic in a lot more detail. It covers, among other things, how to use the Reports workspace to analyze and report on your existing site, or on other external sites, in order to show what needs fixing or improving to make the sites search-engine friendly.

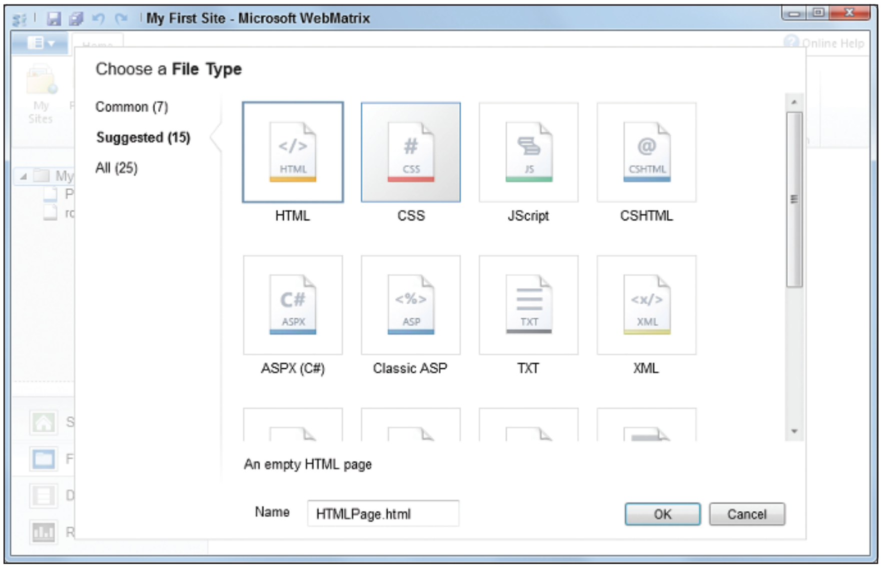
The Reports workspace is unique to WebMatrix. Even the professional ASP.NET development environments — Visual Web Developer and Visual Studio — don't include this fantastic tool!

Common WebMatrix Templates

To make creating new sites and pages as quick and easy as possible, WebMatrix ships with a number of predefined templates for new files and websites. The next section explains the most commonly used file types, followed by an overview of the available site templates.

File Types

During the "Try It Out" earlier in this chapter, you added a new file to your website. At that time, you were presented with the Choose a File Type dialog box (shown in Figure 1-9), which showed a selection of file types.



An empty HTML page

Name: HTMLPage.html

FIGURE 1-9

This section explores the more common file types available to you and explains what they are used for.

HTML

The HTML file type is used for static HTML documents. The content is text based, and served directly by the web server. You will use this file type as part of exercises that are designed to show HTML and CSS. Otherwise, you are unlikely to need this file type for Web Pages development.

CSS

CSS (Cascading Style Sheet) files are used to control the presentation and formatting of web pages. You will learn much more about how to use this kind of file in Chapters 2 and 3.

JScript

JScript is Microsoft's implementation of ECMAScript, which is also the basis of JavaScript. JScript is a client-side (browser-based) scripting language, which is most often used to provide a richer user experience than HTML can offer. For all intents and purposes, JScript and JavaScript are the same thing. For convenience, the term JavaScript will be used throughout this book when referring to JScript files.

CSHTML/VBHTML

Razor syntax can be implemented using either C# or Visual Basic. CSHTML files are ASP.NET Web Pages files that are designed to work with Razor using C#, whereas the VBHTML extension is used for files that contain Razor code using Visual Basic. Note that Razor code can be (and often is) mixed with plain HTML in a CSHTL or VBHTML file.

ASPX (C#/VB), Classic ASP, and PHP

WebMatrix is not just a tool for developing Web Pages sites. You can also develop ASP.NET Web Forms sites with it, as well as sites with Classic ASP or PHP. Web Forms, Classic ASP, and PHP are beyond the scope of this book.

XML

XML is a text-based mark-up language similar to HTML. Its purpose is to add structure to data so that humans and machines can easily interpret it. Data is wrapped in HTML-like tags, which are defined by the creator of the XML file. The most common uses for XML are configuration files, and for transporting data from one platform to another.

Web.Config (4.0)

The Web.Config file is an XML document that contains configuration settings for your ASP.NET Web Pages site. The 4.0 in the title refers to the version of the .NET Framework that the file is set up to use.

Class (C#/VB)

A Class file contains pure C# or VB code. Sometimes you will want to add code to your site that might be used in a number of places. Rather than copying and pasting the code each time you want to use it, you create a routine using pure C# code, and call that routine wherever you need it. You'll see how to make use of class files later in this book.

Besides templates for files, WebMatrix also has templates for complete websites, helping you jump-start your site development.

WebMatrix Site Templates

WebMatrix includes a range of site templates. You have already seen the Empty Site template, which lives up to its name. Now you will explore the other templates and see how they can be used to give you a head start in your website development.

The Starter Site

The Starter Site is just like the Empty Site in that it is blank. It has no theme. Its purpose is to give you a starting point for building a site that makes use of the Web Pages security and membership features, which you will explore in detail in Chapter 12, "Security."

The Bakery Site

The Bakery Site is an example of an e-commerce site. The site content is driven from a database, and the workflow includes a simple online ordering system. You will learn how to use databases to power your site in Chapters 10 and 11.

The Calendar

The Calendar is one of the most advanced sites among the templates and serves as a starter site for an online Calendar application, similar to Outlook Web Access or Google Calendar. It offers features such as adding and viewing events, sharing your calendar with others, downloading event details, and user selectable themes.

The Photo Gallery

The Photo Gallery also includes security and membership and its content is database-driven. The real point behind the Photo Gallery is to showcase some of the built-in WebMatrix features for managing images.

ANATOMY OF A WEB PAGES APPLICATION

You have already been introduced to the types of files you will use most often in your Web Pages development. Using the Calendar as a guide, you will see where the various types of files go, and also learn about some of the special folders offered by WebMatrix. You can either create a new site based on this template in WebMatrix to follow along with this section, or refer to Figure 1-10, which shows the Files workspace for a new Calendar website.

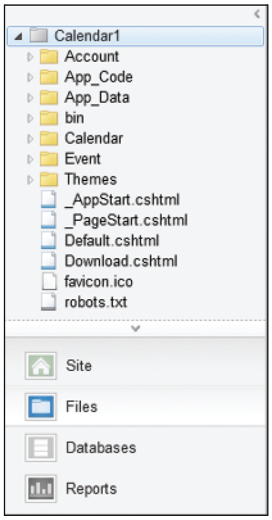


FIGURE 1-10

Default Folders in a Web Pages Application

An ASP.NET Web Pages website can contain a number of special folders, each one briefly described in the following sections.

App_Code

Within ASP.NET, App_Code is a special folder. It is the place where you will add C# or VB Class files containing only C# or VB code.

By default, the web server is configured not to allow users to browse this folder and download its contents. In the Calendar template site, App_Code, contains a number of custom helpers written purely in C#. You will learn more about creating your own helpers in Chapter 4.

App_Data

Like App_Code, the App_Data folder is a special ASP.NET folder. It is also protected from being browsed by users and is primarily used for placing database files. You can put any kind of data in here, including SDF files (for SQL Server Compact), CSV files, XML files, and so on.

Bin

The Bin folder is the place for binary files and compiled assemblies (DLLs). When you use the Package Manager (explored in more detail in Chapter 7, “Packages”), any DLL files will automatically be downloaded and added to the Bin folder. If you find yourself needing to use third-party components, such as those required for managing PDF creation, you will place their library files in the Bin folder. Again, the web server is preconfigured not to serve the contents of this folder to users.

The previous folders all have special meaning in ASP.NET, and their names are important. The following folders are named the way they are by convention. Each of the template sites follows the same naming policy for consistency, but you are free to rename the folders as you like.

Account

When you use the Membership features, you will place your account-management files in the Account folder. In the Calendar sample site, you can see that the files are all CSHTML files, because they need to access a database or perform other server-side tasks. Renaming this folder will require a small configuration change to your site, which you will learn about in Chapter 11, “A Deeper Dive Into Data Access.”

Content

By convention, your images and other media files are placed here, along with style sheets (covered in Chapter 3).

Scripts

JScript files containing JavaScript code are placed in the Scripts folder.

Shared

In most articles and books on WebMatrix, the Shared folder is the location for Layout pages (examined in Chapter 3) and reusable blocks of content.

Although you can rename these folders, I advise against it. There are two reasons for that advice: The first is that following a consistent naming convention makes it much easier to locate items in the future. You may not think so now, but it is very easy to forget how your site is structured when you look at it at some time in the future. You will be thankful that you worked consistently when you are asked to add new features to a site you have long forgotten. The second reason is that the folder structure offered by the WebMatrix templates follows the ASP.NET MVC templates. The upgrade process will be a lot simpler if you don't have to rename your folders to follow the slightly different rules that apply to MVC, where for example, the Shared folder does have special meaning.

The Root Folder

There is one more very important folder in an ASP.NET Web Pages site, and that's the Root folder. You won't see this appear in the file and folder view labeled "Root," but it is the top-level directory in which all other folders and files reside. When you look at the folder structure in the left pane, it is the folder that appears at the top of the listing as a grey icon and is labeled with the name of your site.

SUMMARY

In this chapter you learned what WebMatrix is. You learned that it is composed of four integrated parts — a web server, a database platform, a development framework, and a tool for developing websites using Microsoft technologies.

You also learned how to obtain WebMatrix, and how to download and install it very simply using the Web Platform Installer. You tested that your installation was successful through a simple exercise that announced to the world that you are on your way to becoming a web developer. You were introduced to the various workspaces within WebMatrix, which will help you use the tool most effectively when managing different aspects of web development.

Understanding the relationship between the client and the server is fundamental to becoming a successful web developer. This chapter explored that relationship and explained the important roles played by each party in the relationship. It also covered, in simple terms, how ASP.NET works on the web server, and its role in differentiating between HTML, CSS, and code to generate web pages dynamically.

You explored the types of files you are likely to use most often with WebMatrix, and learned their role in helping you build your site. Finally, you looked at the site templates offered by WebMatrix and explored the Calendar site template to understand how to manage your folders and files within a Web Pages application.

In the next chapter, you will learn more about HTML and CSS, and learn how important these technologies are for building websites.

EXERCISES

1. What's the best way to acquire and install WebMatrix?
2. What's the difference between a CSHTML and a CS file?
3. Name at least two special ASP.NET Web Pages folders and explain what types of files they contain. Besides the types of files these folders contain, can you mention another reason why these folders are special?
4. What's the difference between static and dynamic files?

Answers to the Exercises can be found in the Appendix.

▶ WHAT YOU LEARNED IN THIS CHAPTER

TOPIC	KEY CONCEPTS
WebMatrix	A stack of software components to build web applications. There are four core components to the stack: A web server (IIS Express); a development framework (.NET 4.0); a database platform (SQL Server Compact Edition 4.0); and a lightweight web authoring and management tool. Often, the "lightweight web authoring tool" is simply referred to as WebMatrix.
IIS Express	A lightweight web server designed specifically to be used during development.
SQL Server Compact Edition 4.0	A file-based database system used to store data.
Web Platform Installer (WPI)	The unified download tool for many of Microsoft's developer products.
Razor syntax	A simple programming syntax for embedding server-based code in a web page. It enables you to mix plain HTML and server-side logic in a very succinct way.