Getting Started with Dreamweaver MX

Over the past twenty years the Internet has evolved tremendously, from an obscure networking tool used by select academic and government institutions to a means of connecting millions of people worldwide.

Yet the Internet is still, in its most basic form, a **wide area network**. This means that it is a series of interconnected nodes that can connect computers together over a large (or even global) geographic area. Today, at any given moment, the number of people using this wide area network is estimated to be between 600 million and 1 billion; and this number continues to grow. It is just as important to people who use it simply to say "Hello" to a friend many miles away, as it is to major international corporations who use it to connect offices worldwide.

With the growth in usage and sophistication of the Internet, there has been a corresponding evolution of technological capabilities. With this evolution, new programming technologies have emerged. At one time, we constructed our web pages simply using a "formatting" language called HTML. Now, while HTML is just as important as it always was, we *also* use a host of other technologies (such as scripting languages, ASP, ColdFusion, XML, and so on) to meet these new levels of sophistication.

The great thing is that you don't need to know every last detail about all these technologies to start building attractive, functional websites. There are a number of development environments around that will help you to do the job. Macromedia's contribution to this arena is its **Dreamweaver** development environment. The most recent version, **Dreamweaver MX**, is integrated with other products in Macromedia's MX suite: Flash MX, Fireworks MX, and so on.

What's great about Dreamweaver MX? If you want, you can use it to do the bulk of the coding work for you – that's great when you're not too familiar with the underlying technologies, or when you're trying to get the job done quickly. Moreover, as you become better acquainted with Dreamweaver, with HTML, and with other related technologies, you'll see that Dreamweaver also allows you to tinker with the code itself – or simply to look at the code and get a better understanding of what you're creating.

As exciting as all this sounds, we must start at the beginning. Before we can do anything we must set up a few things. So, in this chapter we will:

- Take a quick look at how the Internet works, and how Dreamweaver (and other tools from Macromedia) play their part in the creation of websites
- Install the tools you will need as you work through this book – the Dreamweaver MX web development tool, and the Microsoft IIS web server
- Set up the Dreamweaver workspace to suit your own personal needs

Over the course of the first eight chapters of this book, we'll build a website on a subject close to my heart – food and cookery. While we indulge my passion for fine cuisine, we'll also gain some familiarity with the Dreamweaver development tool, and begin to learn about the art of web development with Dreamweaver MX.

As we progress through the book, we'll also learn a lot about HTML, scripting languages, and ASP, and we'll see how to use Dreamweaver MX to help us create great websites using these important technologies.

A Brief Overview of the Internet

I'm sure you're familiar with using the Internet. You open a browser; you type in the address of a website (or click on a button or a link); and then you wait a while. Then, after a short time, a page appears in the browser. It's simple.

But before you can request that web page, somebody somewhere has to take the trouble to create the web page and publish it. (They may even have used Dreamweaver MX to do that.)

In this section, we'll look at this simple process, and get an understanding of how the page request is processed, because it's at the root of everything else we do in this book.

Static Web Pages and Dynamic Web Pages

When it was first born (and for a while even after it started to become more widely used), all of the web pages published on the Internet were **static** web pages. A static web page, essentially, is a web page that is created, in its entirety, at some time before any user visits the page.

Even now, there are still plenty of static web pages out there on the Internet. They might contain text, graphics, images, hyperlinks (links to other pages), and so on. The thing about a static web page is that it always looks the same, regardless of who visits the page, when they visit it, or how they arrive at the page. The only time a static web page changes is when the web designer (or webmaster or web author) physically locates the web page file, opens it, and edits it.

Hypertext Markup Language (HTML)

Static web pages are composed using a language called **HyperText Markup Language**, or **HTML**. HTML is a **markup language** – its purpose is to describe both the *content* of the page (the text, and the graphics used) and the *layout* and *positioning* of all the elements of the page. Here's a simple example:

```
<html>
  <head>
    <title>Welcome to Beginning Dreamweaver MX</title>
  </head>
  <body>
    <h1>Welcome!</h1>
    <p>
      This is the first example in Wrox's <i>Beginning Dreamweaver MX</i>
      book. To learn more about Wrox books, go to
      <a href="http://www.wrox.com">www.wrox.com</a>.
    </p>
  </body>
</html>
```

As you can see, HTML is composed of **tags** (the things enclosed in <...> characters) and text. The tags tell the browser how to display the text. For example:

- The phrase `Welcome to Beginning Dreamweaver MX` sits between a `<title>` tag and a `</title>` tag, which tells the browser that this phrase should be displayed in the title bar.
- The phrase `This is the first example...` sits between a `<p>` tag and a `</p>` tag. These are paragraph tags, so this tells the browser that this phrase should be displayed as a paragraph.

This HTML would usually be contained in an `.htm` or `.html` file. When this web page is displayed in a browser, it looks something like this:

As you can see, the HTML shown above also describes:

- The placement and format of the header Welcome! (in a larger font)
- Some italicized text (between `<i>` and `</i>` tags)
- A hyperlink www.wrox.com (which also known as an anchor, which is why HTML uses `<a>` and `</a>` tags to describe it).

There are different ways that the author of a web page can generate the necessary HTML:

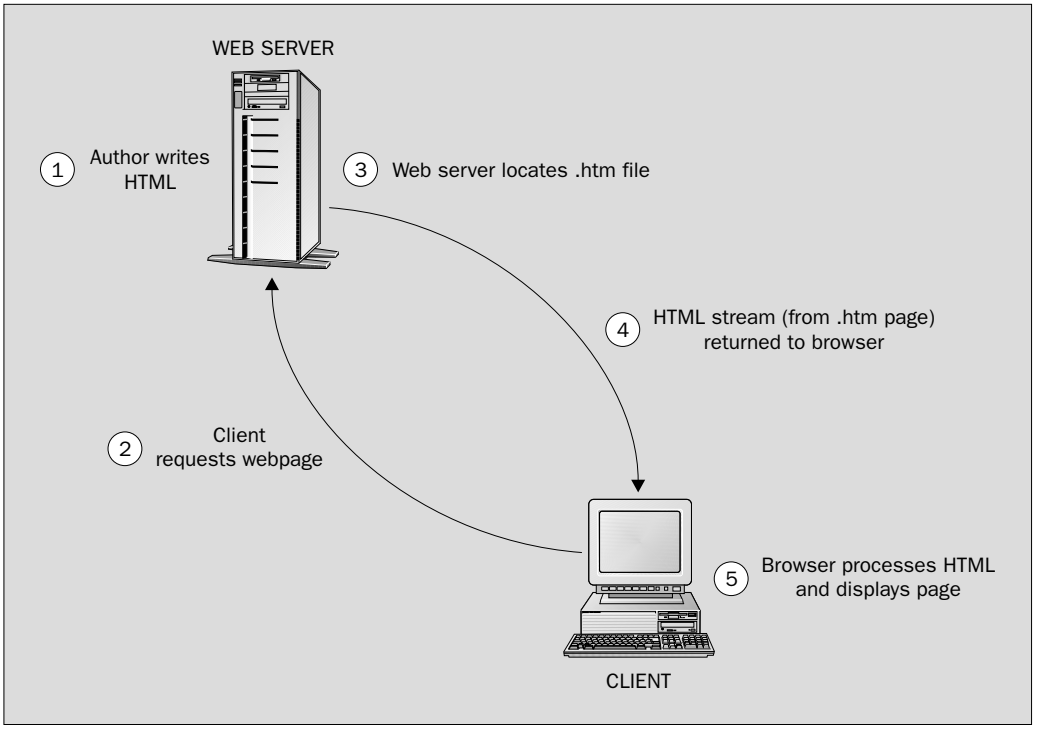
- They may choose to type the HTML into a text editor such as Notepad
- Perhaps more likely, they may choose to design and compose web pages using a product like Dreamweaver, which (if they wish) will generate the HTML for them

Don't worry too much about the finer details of HTML now. We will learn as much as we need about HTML tags as we progress through the book. The main question for now is that of how the HTML gets to the browser.

How the Page Gets to the Browser

The one thing you should note about that page is that its content and layout are *completely* determined by the HTML you see above. So, no matter who visits this page, or when they visit it, it'll always look the same – unless the author of the page rewrites it.

So how does a static web page get displayed on a browser? It's about clients and servers, and it's about requests and responses. The following diagram illustrates the entire process:



There are two machines in this diagram; a server and a client:

- The **client** machine is the machine on which a user is running a web browser, and with which they click a button to request a web page
- The **server** machine is a machine that runs the web server software – the software that manages the web pages and makes them available to client machines (via the Internet or via a local network)

In Step 1, an author composes the HTML file, saves it, and publishes it on a web server. At any time after that, a user (using a browser on a client machine) types a web address into their browser, or clicks the button, to make a **request** for the page (Step 2).

This request travels across the Internet (or network) and arrives at the web server, which locates the `.htm` file that was requested (Step 3), and sends it back to the client machine (Step 4) as a **response**. If the HTML refers to any image files, then they are also sent as part of the response.

Finally, when the browser receives the response, it interprets the HTML tags and text, and displays them on the screen as instructed by the HTML (Step 5).

Dynamic Web Content

As the Internet evolved, the limitations of static HTML static pages became apparent. Web authors could only write pages that always look the same, which meant that there were all sorts of things that couldn't be done with static web pages:

- They couldn't write a page that displayed the name of whoever was looking at it
- They couldn't write a page that reflected the latest conditions and environment, such as the current time, or the latest weather
- They couldn't write a page whose content was dependent on what the user had chosen in the previous page they looked at

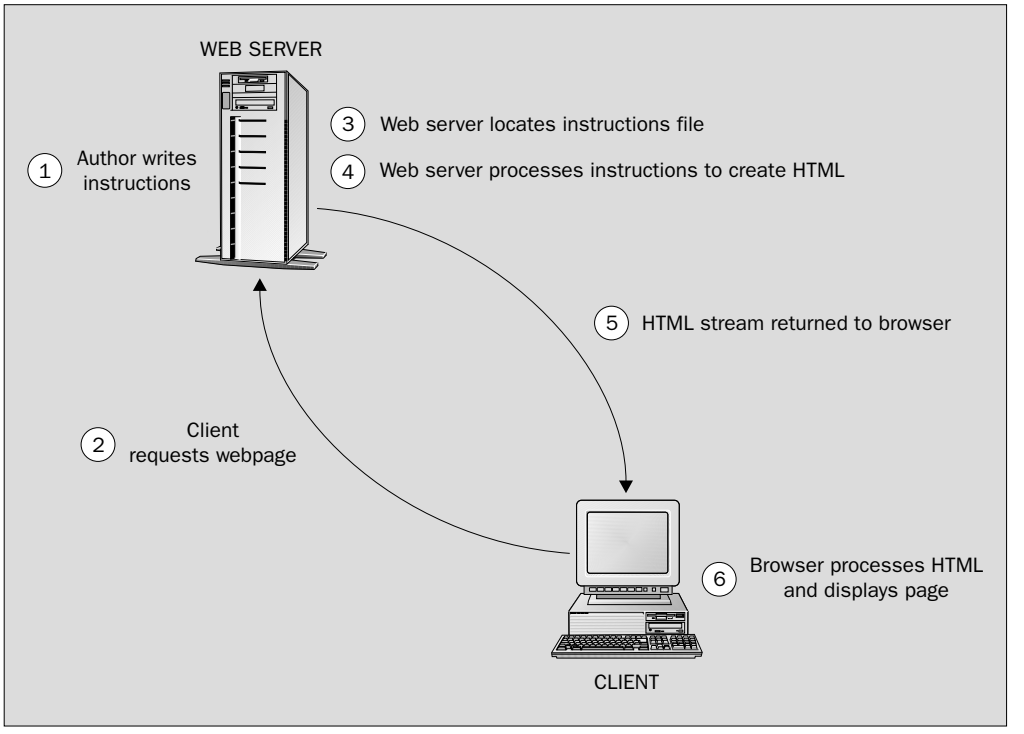
Web authors needed something more flexible, and so **dynamic web content** evolved. A **dynamic web page** is a web page that can contain both HTML *and* instructions for generating HTML. When a user requests a dynamic web page, the response they get is still HTML; however, some (or all) of that HTML is generated by software after the user requested the page, and can therefore **react** to the user's actions, to the current situation, and to the latest information that is available.

Server-Side Processing

For example, suppose that you want to see a list of books, written by Ernest Hemingway, on Amazon.com. To do this, you use a browser to browse to Amazon's website, you type the word Hemmingway into a textbox, and you click a button to submit this page request. Now, Amazon has information on millions of titles – so how does Amazon's web server know what information to show you?

The answer is that the web server analyses the information that your browser sends when you request the page, and detects that you typed the word Hemmingway into that textbox. It uses this information to perform some processing, and to generate HTML that describes a web page full of information about the Hemmingway books in Amazon's catalog. When that HTML is generated, it is sent back to your browser. Then your browser interprets the HTML and displays the page, just as it does for static pages.

So, in other words, this web page is generated *dynamically*, by the web server, at the time it's requested. We can compare the request/response process here with the process for a static page. As you can see, there's only one extra step (Step 4):



Now, as we've already noted, HTML is just a markup language, and it's only good for describing the formatting and layout of web pages. We need something else – something that is capable of describing the processing instructions required to generate HTML code when it's requested. To fulfill this requirement, a procedural processing language is required. To begin with, website developers and web pages authors used languages such as **CGI** and **Java** on the web server. These days, **scripting languages** such as **VBScript**, **JavaScript**, and **PHP** are also very commonly used on the web server, and their capabilities are augmented with technologies such as **JavaServer Pages** (**JSP**), or Microsoft's **Active Server Pages** (**ASP**), or Macromedia's **ColdFusion**, which provide further flexibility and web-specific functionality (as we'll see in Section 2 of this book).

In fact, in this book we'll be using VBScript, JavaScript, and ASP for our server-side processing. However, that will come later – beginning at Chapter 7.

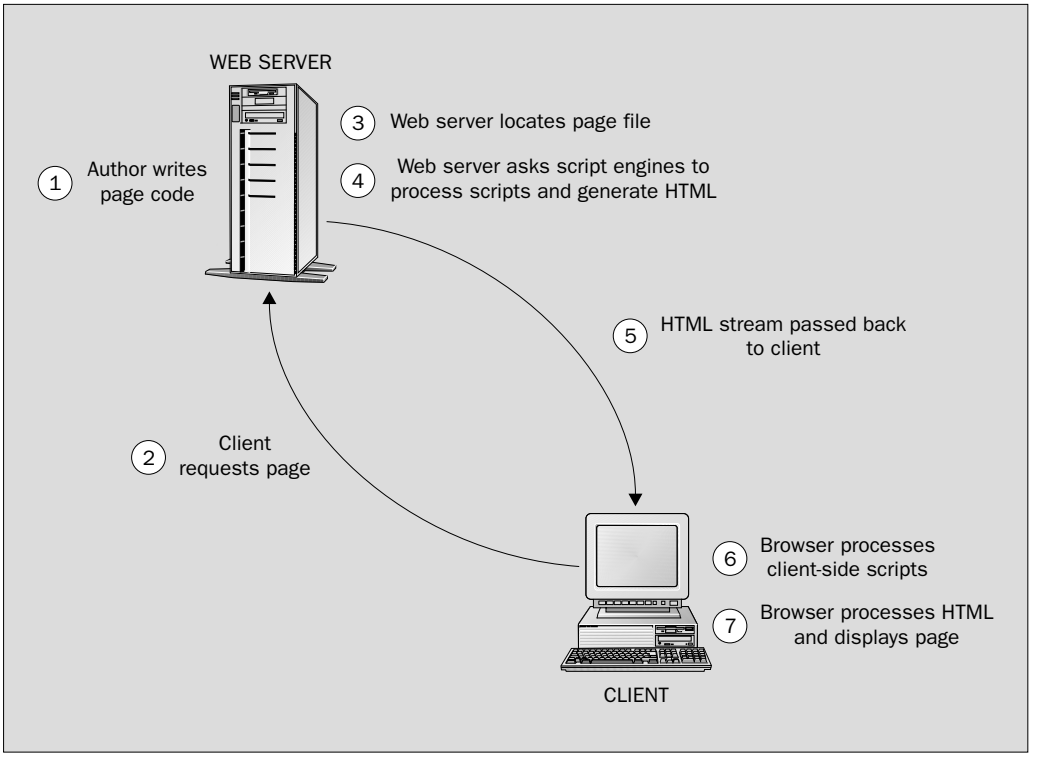
Client-Side Processing

We've talked a little about using a language to write processing instructions (lines of code) that are executed by the web server, and how that technique can be used to generate HTML. We can also write code to be executed at the *client*, and the effect of this is quite different.

Client-side code is not executed by the web server. Instead, it is sent to the browser (along with HTML, as part of the response), and is processed by the browser; and the browser displays the results on the user's screen.

By using client-side code, we can bring the page to life, so that the page reacts to the user's button-clicks and mouse-movements. This means that the page can display attractive and functional characteristics, even though the page has left the web server.

So, we can extend our picture a little further, adding one more step (Step 6) to take care of the client-side processing:



Of course, if our client-side code is to work, then the client machine (or its browser) must support the language in which it was written! JavaScript is probably the most commonly used language for writing client-side code as it's supported by all the major browsers (Internet Explorer, Netscape Navigator, Opera, and so on). As we'll start to see in Chapter 2, when we use Dreamweaver to create dynamic client-side effects, it generates JavaScript client-side code to power these effects.

Dreamweaver MX

In the summer of 2002, Macromedia released its **Studio MX** development environment. The MX environment was built to address the development issues associated with newer technologies. In addition, MX incorporates the experience and suggestions of thousands of developers – who have helped to fine-tune its features to reflect the needs of modern high-productivity web development. Studio MX is an integrated suite of development tools, consisting of:

- **Flash MX**, for delivering sophisticated animations and more efficient websites
- **Fireworks MX**, for developing exciting, attractive graphics
- **ColdFusion MX**, for providing server technology for the delivery of dynamic web content
- **Dreamweaver MX**, which is the principal development tool for bringing all the other components together

This book focuses on the Dreamweaver component of this suite. As we'll see as we progress through the book, Dreamweaver MX offers a large variety of tools for design, coding, and server integration:

- It gives the relatively non-technical developer access to the "clever bits" of web development, by allowing the development of quite complex applications with little or no knowledge of programming languages or databases
- It also gives the more experienced or technically-minded developer an environment in which they can create websites quickly, because it allows these users to put together the elements of their site without writing every single line of code

Ultimately, Dreamweaver greatly enhances our ability to produce attractive, bug-free products in an environment where development time is at a premium.

Installing Dreamweaver MX

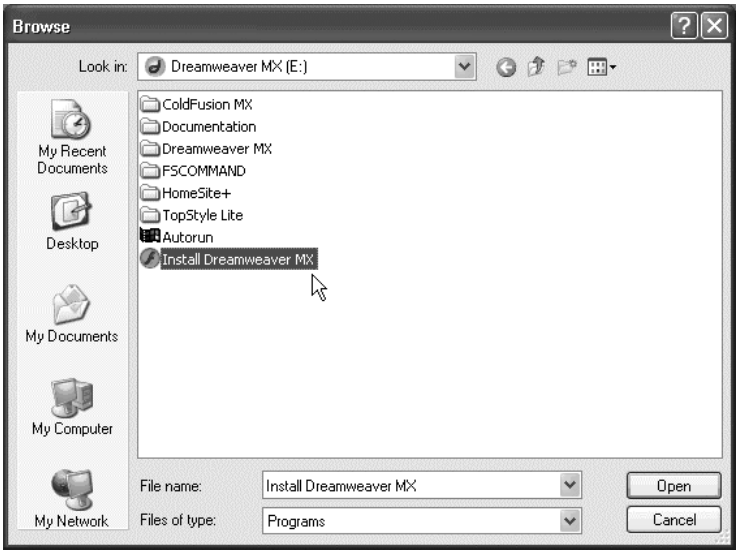
There are several ways to obtain Dreamweaver MX. You can download a 30-day evaluation from http://www.macromedia.com. You also can purchase it – by itself, or bundled with the Macromedia Studio MX suite. The installation process for all of these versions is largely identical – there are just a few minor differences.

Let's step through the Dreamweaver MX installation process. The following screenshots were taken from the Macromedia Dreamweaver MX installation disk, using the Windows XP Professional operating system. If you're installing onto a different operating system, or from a different source, then you might see one or two differences, but the overall process should not vary hugely.

Try It Out **Installing Dreamweaver MX**

1. Run the installation program. If you're installing Dreamweaver from a CD-ROM, then the installation program will probably run automatically when you insert the install disk.

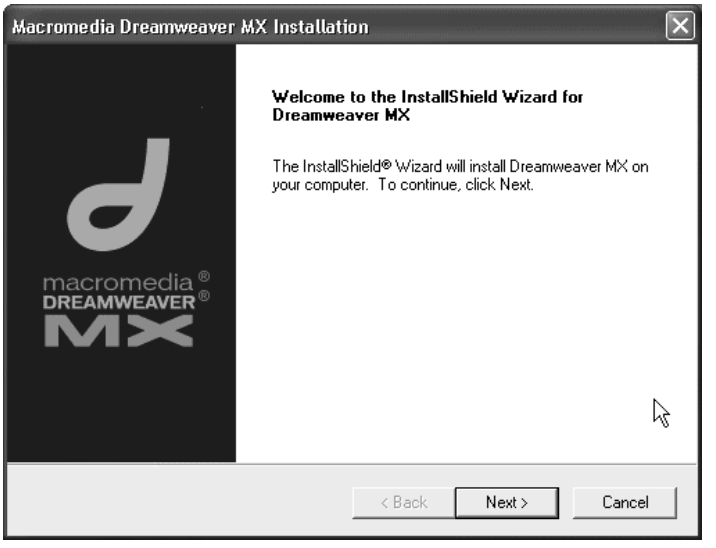
   If not, select Start | Run, and then click the Browse button and browse to the Dreamweaver MX Installer (called Install Dreamweaver MX or Install Dreamweaver MX.exe):



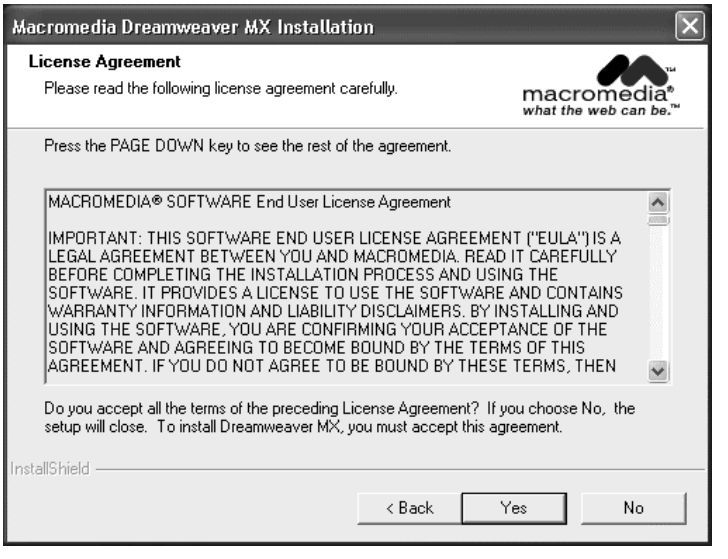
   Then click the Open button to select that file and then the OK button to run it.

2. You may see a Flash graphic, inviting you to choose which product you want to install. If you do, select Dreamweaver MX.

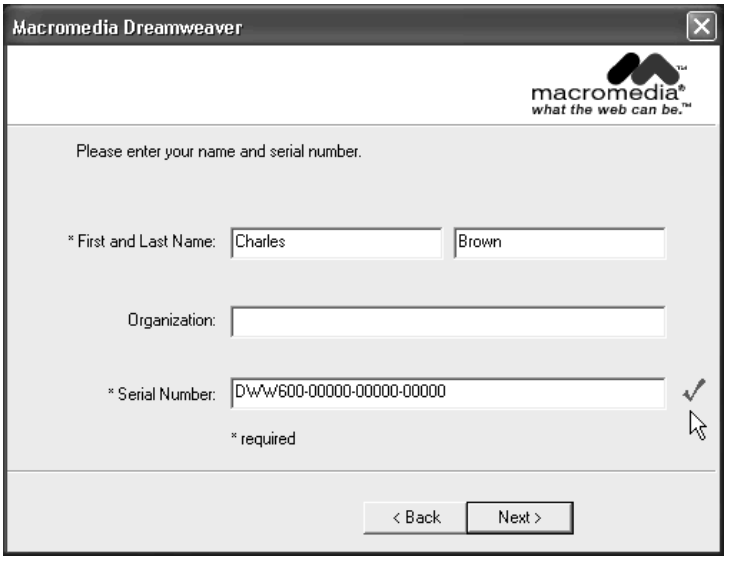
3. You'll see the Wizard begin, and you'll have to wait a short time while the source files are extracted. Then you will see the following screen. Click Next:



4. Next, you'll see the page that asks you to read and accept the License Agreement. To accept it, click Yes:

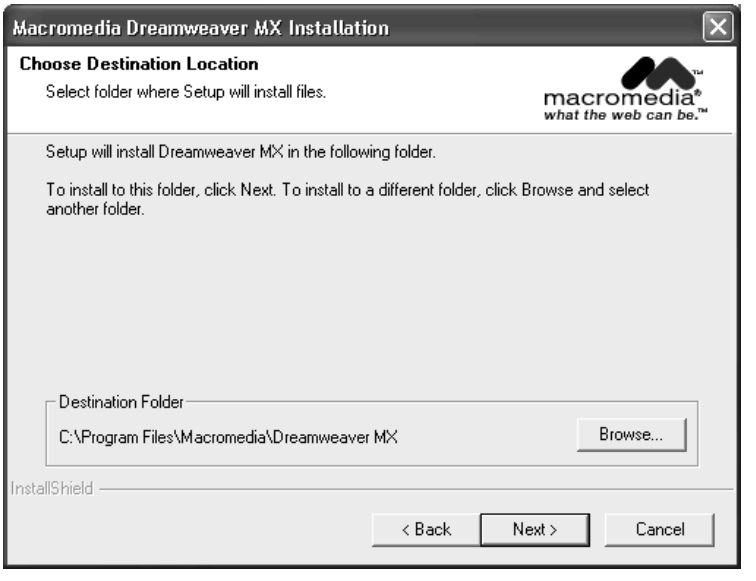


5. Next, the Wizard asks you to enter your name and the serial number of your disk. Type them in (when you type in the correct serial number, the Wizard confirms it with a green check). Then click Next:



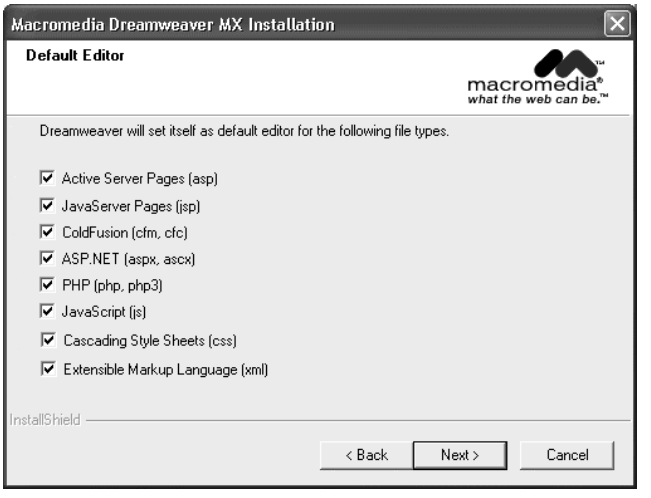
6. If you're upgrading from an older product, you'll be asked to enter the serial number of that product too. Do that, and click Next.

7. Next, you have a chance to change the installation directory. It's usually sufficient simply to accept the default option here. (You can install it into a different location, if you want. To do that, use the Browse button.) Click Next.

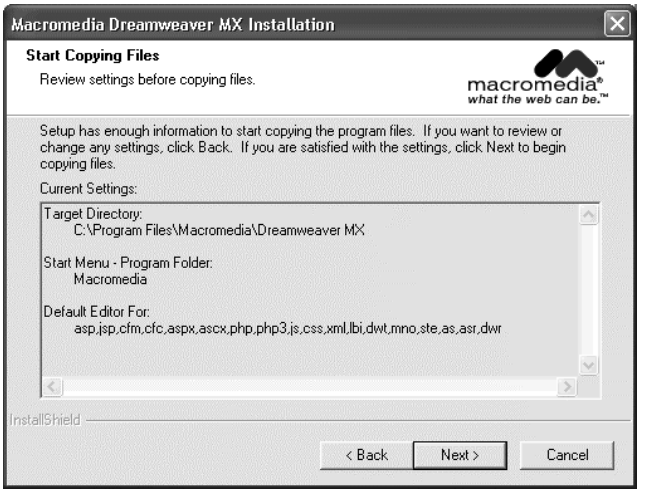


8. In the next screen, you'll see a list of programming languages and other technologies. You can use Dreamweaver MX to edit code written using any of these languages and technologies. (As you can see, Dreamweaver is a powerful development tool!)

Use the checkboxes to indicate which technologies you plan to work with in Dreamweaver. By checking these checkboxes, you're effectively telling your machine to use Dreamweaver whenever you edit a file of one of these types. For now, leave all of these checkboxes checked (you can always change this configuration after you've installed Dreamweaver). Then click Next:



9. The Dreamweaver MX Installer will present a confirmation page, to tell you what options you've selected:



Click Next to begin the installation process. You'll need to wait a while as the Wizard completes the installation. When it's finished, the Wizard will display a window saying that Setup has finished installing Dreamweaver MX on your computer. Click Finish to close the Wizard.

How It Works

This process installs all the necessary files and performs the configuration required to get Dreamweaver working. When it's complete, the Dreamweaver development environment will be ready for us to use – if you want, you can start up Dreamweaver from the Start menu, using Programs | Macromedia | Macromedia Dreamweaver MX. We'll come to that in a moment, but first there is something else we must attend to – the web server software.

By installing Dreamweaver MX, you have installed a powerful code editor. However, that would be using only a small part of the Dreamweaver capability. A major feature of this program is its ability to integrate with server technologies – which is particularly useful for rapid web application development tasks.

Rapid Application Development, or **RAD**, is programming jargon – essentially, it equates to "getting the code out as fast as possible". Some development tools, such as Dreamweaver MX, help the RAD process by generating bug-free code as you use visual tools to put the text, graphics, database integration (and so on) into place. All we do is **drag and drop** the various components of our web page to the place we want them, and the development tool generates the HTML and scripting code for us. We do not have to spend long and frustrating hours debugging.

Installing a Web Server

As you've probably guessed from the discussion earlier in this chapter, we're going to need a web server in order to be able to build the websites in this book. The choice of web servers available to you is rather dependent on which operating system you're using:

- If you're using **Windows 2000** (Professional, Server, or Advanced Server editions), or **Windows XP** (Professional Edition), then perhaps the easiest thing to do is install Microsoft's **Internet Information Server** (**IIS**) web server software – we'll step through this process in a moment.
- If you're using any other operating system, then you need to consult Appendix E (or the Dreamweaver documentation) to find out more about how to use a remote web server. Discussion of the various possible combinations is beyond the scope of this chapter.

Installing Microsoft's IIS Web Server

In this book, we're going to use Microsoft's **Internet Information Server** (**IIS**) web server software, which is bundled with the versions of Windows 2000 and Windows XP listed above. We're going to install IIS onto the same machine as our Dreamweaver installation but you don't have to do that. If you prefer, you can use two machines – one with Dreamweaver installed, and the other with IIS installed.

> *Later in the chapter, we'll start up Dreamweaver and prepare it ready for us to start writing websites on it. As part of that process, we'll tell Dreamweaver about the IIS installation (or web server software) that we want to use to test the site we're building.*

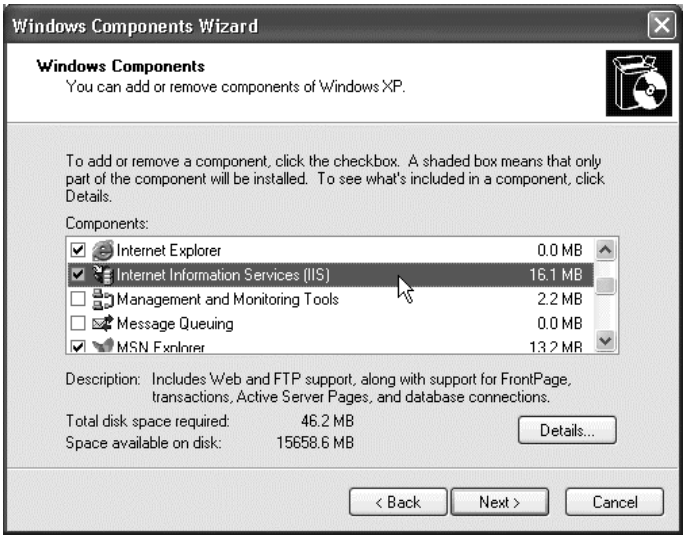
IIS version 5.0 is supplied as part of Windows 2000, while Windows XP Professional ships with IIS version 5.1. For our purposes, there is not much difference between IIS 5.0 and IIS 5.1, so whichever version is supplied with your operating system will do.

Try It Out — Installing and Testing IIS 5.x

There are several ways to install the IIS web server software. If you are using Windows 2000 or XP Professional, just follow this simple procedure.

1. Launch your Windows 2000 or XP installation CD. Select the Install optional Windows components option:

2. You'll see the following dialog. If the Internet Information Services (IIS) checkbox is already checked, then IIS is already installed (so you Cancel this, and skip to Step 4 to test it). Otherwise, check this checkbox and then click Next:

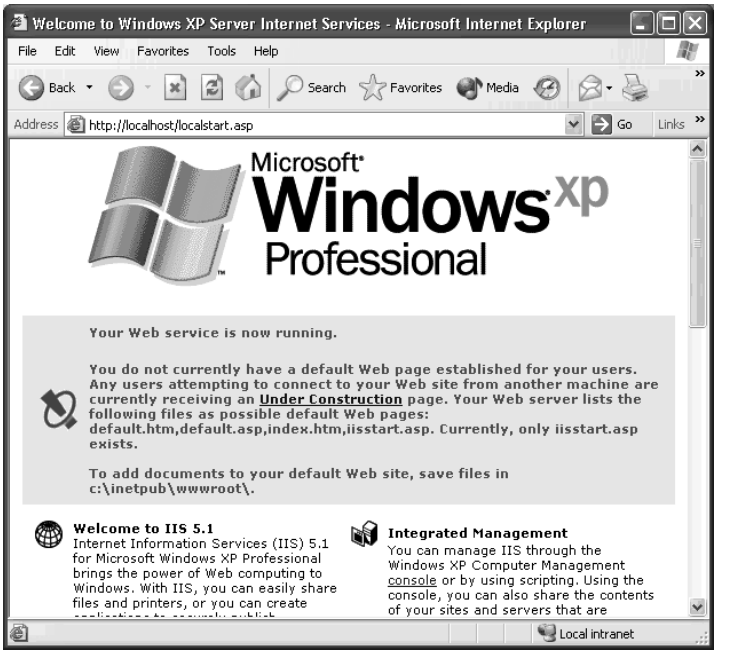


3. Click Next. The Wizard will install and configure the components of IIS; as it does so, it will display a status bar to inform you of the progress of the installation. When it has finished, you'll see a message that tells you that you have "successfully completed the Windows Components Wizard", and you can click Finish to close the Wizard.

Note that you don't have to start the IIS installation from the operating system CD. Instead, you can get to the dialog in Step 2 by selecting Start | Control Panel *in Windows XP (or* Start | Settings | Control Panel *in Windows 2000) and then clicking* Add or Remove Programs *and then* Add/Remove Windows Components*. From there, you can follow Steps 2 and 3 – when you reach Step 3 you'll be asked to enter the operating system CD or specify the place on your network where the installation files exist.*

4. It's a good idea to test the IIS installation now, to ensure that ASP is running. We can do this using the same machine onto which you've just installed IIS.

Using that machine, open the web browser and type http://localhost/localstart.asp into the Address box. It should display the following page:



How It Works

Once you've installed the IIS web server software, it will start running automatically and will continue to run in the background, listening for requests for web pages.

In Step 4, we used a web browser to make a request for a web page at an address of http://localhost/localstart.asp. There are three important elements to this address, and we'll look at them in reverse order:

- At the end, localstart.asp is the name of the web page that we requested. It doesn't have an `.htm` extension; it has an `.asp` extension, and in fact it uses the ASP server-side technology we mentioned earlier. ASP is installed as part of IIS, so the page should work just fine.
- Before that, the bit just after the // indicates the location of the web server on which we expect this page to be hosted. Here, we've specified localhost, to indicate that we're trying to contact the web server that's on the same machine as the browser.
- Before that, the address begins with http://. This causes the web page request to be transported from the browser to the web server using something called **Hypertext Transfer Protocol**, or **HTTP**. HTTP is the protocol used by the Web to describe both requests and responses. You can think of it as a standard format for web requests and web responses. Because it's a standard, all web servers and browsers and other machines on the Internet can understand the request or response message being sent.

In this case, the browser and the web server are on the same machine but that doesn't have to be the case. If you have a network set up, then you should be able to browse to the `localstart.asp` page on your IIS web server from other machines on your network (you'd use an address like http://*<iis-machine-name>*/localstart.asp).

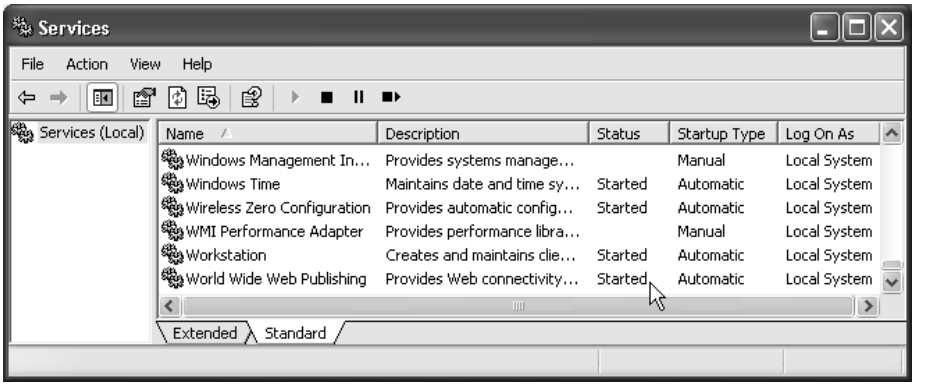
If your IIS machine is exposed on the Internet, then you may also be able to browse to the `localstart.asp` page that way too.

Be particularly sensitive to security risks if your web server is exposed to the Internet. There are plenty of precautions you can take. For example, shut down the server when you are not developing; and download the security patches, using Windows Update (http://windowsupdate.microsoft.com), as they become available.

Simple Troubleshooting

If everything is working properly, your browser should display a test page like the one we've shown in the screenshot above. If you don't get this page, don't panic; it could be that something very simple is wrong. Here are a few suggestions of things you can try to rectify the situation:

- If your browser displays the messages The page cannot be found, and HTTP 404 File Not Found, then it means that your page request reached the web server, but that the web server couldn't find the page you were looking for. Perhaps you misspelled the name of the page – perhaps by typing something like http://localhost/localstaaart.asp. Check and try again.
- If your browser displays the messages The page cannot be displayed, and Cannot find server or DNS error, then it means that it means that your page request did not reach the web server. In this case, it could be that you misspelled the name of the web server in the page address – perhaps by typing something like http://localhosssst/localstart.asp. Check and try again.
- If you've checked the address and you're still getting the same problem, it could be that installation process failed to **start** the web server. To check this, select Start | Control Panel | Administrative Tools | Services in Windows XP (or Start | Settings | Control Panel | Administrative Tools | Services in Windows 2000), and in the resulting dialog, look for the World Wide Web Publishing item:



In the Status column, you should see the word Started, which indicates that the web server is running in the background of your machine, and listening for page requests. If not, right-click on World Wide Web Publishing and select Start to start the service.

If you continue to have problems with your IIS installation, or any other aspect of the book, you should contact Wrox Support at support@wrox.com.

Getting Started with Dreamweaver MX

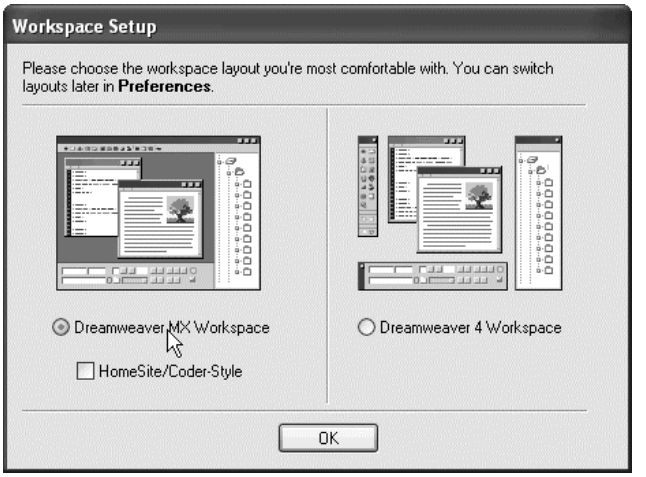
Provided that everything installed properly, you should now be ready to start putting it to work. In this section we'll use Dreamweaver to build a very simple static web page, and then to test it out by viewing it in a web browser to see what it looks like. Then, we'll take a short tour around the Dreamweaver workspace, and introduce ourselves to the different features that we see in Dreamweaver when we first start it up. Once we've completed this example, and our little tour, we'll know enough to get stuck into our cookery-based website in Chapter 2.

The example we'll build is the Welcome web page that we saw earlier in the chapter. You don't need to worry about the little details here – the main purpose of this exercise is just to start using Dreamweaver a little.

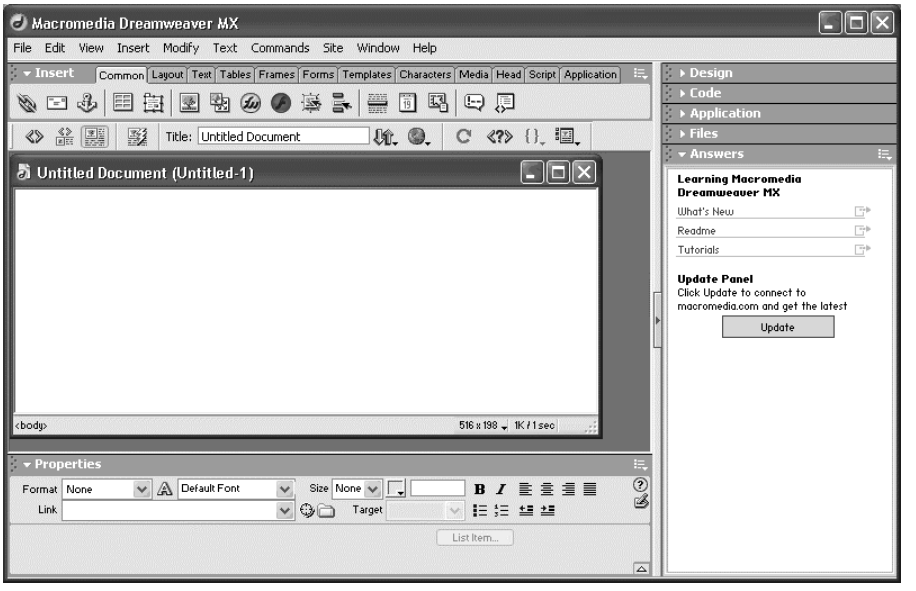
Try It Out — Building our First Example Web Page

1. If you haven't done so already, start up Dreamweaver MX from your Start menu (this should be Start | (All) Programs | Macromedia | Macromedia Dreamweaver MX).

2. If this is the first time you've opened Dreamweaver since you installed it, you should see a Workspace Setup dialog like the one shown overleaf. If so, click the Dreamweaver MX Workspace option, and then click OK (if not, don't worry; we'll deal with you in Step 4):

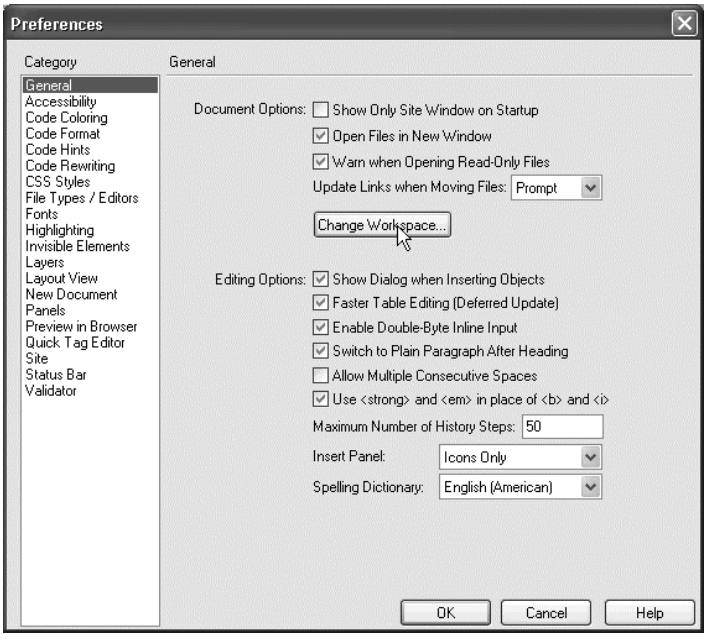


3. Now, you should see the Dreamweaver MX screen:



This is the place where we'll do most of the development in this book. As you can see, there's quite a lot in this window; we'll take a tour and begin to learn about all the different features before the end of the chapter.

4. If you didn't see the Workspace Setup dialog when you opened Dreamweaver, you can open it now. To do this, first select Edit | Preferences from the menu bar at the top of the Dreamweaver window. This will bring up the Preferences dialog, which looks like this:

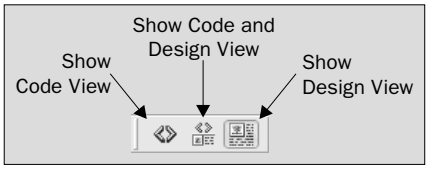


Ensure that the General category is selected on the left, and then click the Change Workspace... button (as shown above). This will give you the Workspace Setup dialog shown in the screenshot in Step 2. Click the Dreamweaver MX Workspace option, and then click OK; and then click OK once more to leave the Preferences dialog.

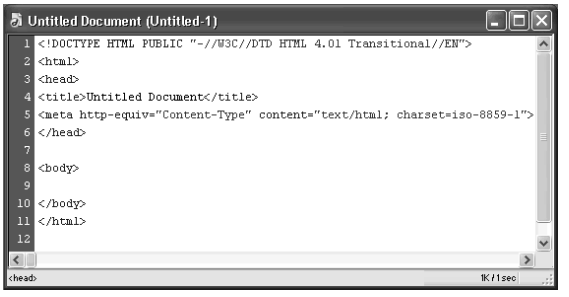
If you changed the setting in the Workspace Setup dialog (in either Step 2 or Step 4), then you will need to close Dreamweaver and restart it in order for the change to take effect. You can do that now, just to be certain.

5. Now we're ready to create our first web page. We could build this page using Dreamweaver's powerful features for designing and building web pages, but to keep things brief in this first chapter, we'll just type the necessary code instead (we'll see plenty of those page design features in the rest of the book).

Near the top left of the window, under the Insert caption (which we'll learn about later), you will see three buttons arranged as shown below. If you float your mouse pointer over the three buttons, Dreamweaver will tell you what the buttons are for:



Click the Show Code View button. You will see the document pane in the middle change to the Code View, and you'll see that the Code View allows us to see the code that describes the web page:

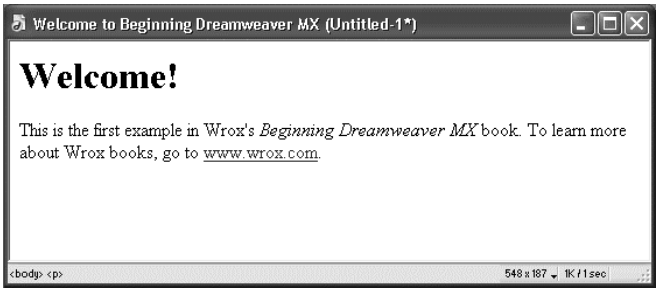


6. Change the code in this pane so that it reads as follows:

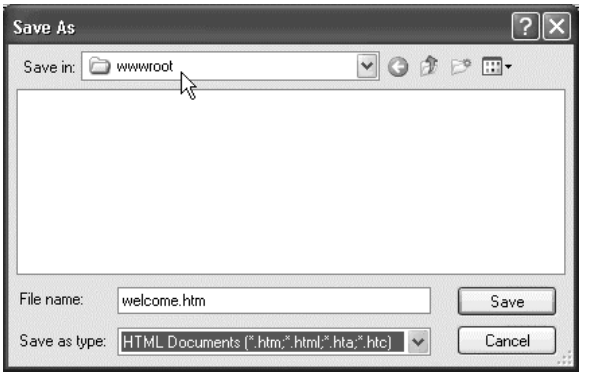
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Welcome to Beginning Dreamweaver MX</title>
  </head>
  <body>
    <h1>Welcome!</h1>
    <p>
      This is the first example in Wrox's <i>Beginning Dreamweaver MX</i>
      book. To learn more about Wrox books, go to
      <a href="http://www.wrox.com">www.wrox.com</a>.
    </p>
  </body>
</html>
```

You'll only have to change the lines highlighted in gray here. Alternatively, you can copy-and-paste the code from the file called `welcome.htm`*, which is contained in the downloadable source code package for this book, available from* http://www.wrox.com.

7. Now click the Show Design View button. In Design View, Dreamweaver allows us to preview our work; more importantly, we can use this view to design pages (which means that most of the time we don't have to deal with code if we choose not to):



8. Now we'll save this file. From the menu bar at the top of the window, select File | Save As, and then use the Save As dialog to navigate to the root folder of your web server (if you have just installed IIS, this should be c:\inetpub\wwwroot). Save the file with the name welcome.htm:



9. Now we can test the page by viewing it in a web browser. Dreamweaver has an easy way to do this, which you'll probably find very useful. Simply press *F12*. This will cause a browser window to appear, and your page to appear in it:

10. It looks swell, doesn't it? However, there is one small issue here. Look at the address displayed in the Address box in the browser: it doesn't begin with http://. This means that Dreamweaver did not make the page request through the HTTP protocol; and *that* means that it is the *operating system*, not the *IIS web server*, that responded by displaying the page in a browser.

    This happens only because we have not yet configured Dreamweaver to set up a test web site, from which it can request pages through HTTP. We'll do that in Chapter 2. For now, if you want to request this page from the *web server*, you can do it without Dreamweaver by typing the address http://localhost/welcome.htm into the Address box of the browser:

How It Works

The Workspace Setup dialog (that we saw back in Steps 2 and 4) simply enables us to control the layout of the Dreamweaver **workspace**. The workspace is simply the user interface – the entire Dreamweaver window and all the buttons and gizmos that you see there. It's a term that we'll use a lot in this book.

The options here are designed for users of *earlier* versions of Dreamweaver. In earlier versions of Dreamweaver, the workspace was organized in a dramatically different way to Dreamweaver MX's workspace, and Macromedia has recognized that users who are familiar with that look and feel might prefer their Dreamweaver *MX* workspace to be organized in the same way. If you're familiar with Dreamweaver 4 here then you could choose the Dreamweaver 4 Workspace option here; but we have chosen the Dreamweaver MX Workspace option and we'll stick with that throughout this book.

We examined the HTML code for this page earlier in the chapter, so we won't do it again here. It is worth noting the two views that we used in the exercise, though:

- The Code View, which allows us to view and edit the code that makes up our web page
- The Design View, which allows us to view and edit the design of the page

By switching from Design View to Code View, adding some code, and switching back again, you will have noticed the relationship between these two views. If we make a change in one of these views, then the change will affect whatever is in the other. That's because these two views are different views of the *same* page.

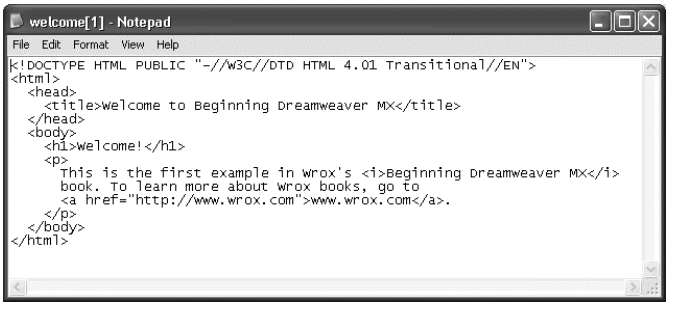
> *From Chapter 2 onwards, we will begin to make much more use of the Design View, and the design tools of Dreamweaver, to build our pages.*

After we saved the file, we viewed it in the web browser. In order to reflect what happens with real-life web pages, we should view the page by requesting it through HTML; we'll return to this subject in Chapter 2.

For now, it's worth understanding what happened, by relating it to the 5-step process we described earlier in the chapter:

- For Step 1, we wrote the page and saved it to disk, calling it `welcome.htm` and placing it in the root folder of the web server.
- For Step 2, we use a web browser to make an HTTP request to the page – by typing http://localhost/welcome.htm into the browser's Address box.
- For Step 3, the IIS web server software received the request, read it, and worked out that the browser's request (for the page http://localhost/welcome.htm) can be answered by returning the page stored at c:\intepub\wwwroot\welcome.htm.
- For Step 4, the web server responds by sending the HTML from this page back across HTTP to the browser (this is the HTTP response).
- For Step 5, the browser receives the response and renders it on screen.

If you want, you can even see the HTML that was sent from the web server to the browser. To do this, right-click on the browser window and select View Source (or View Page Source or similar, depending on what type of browser you're using):



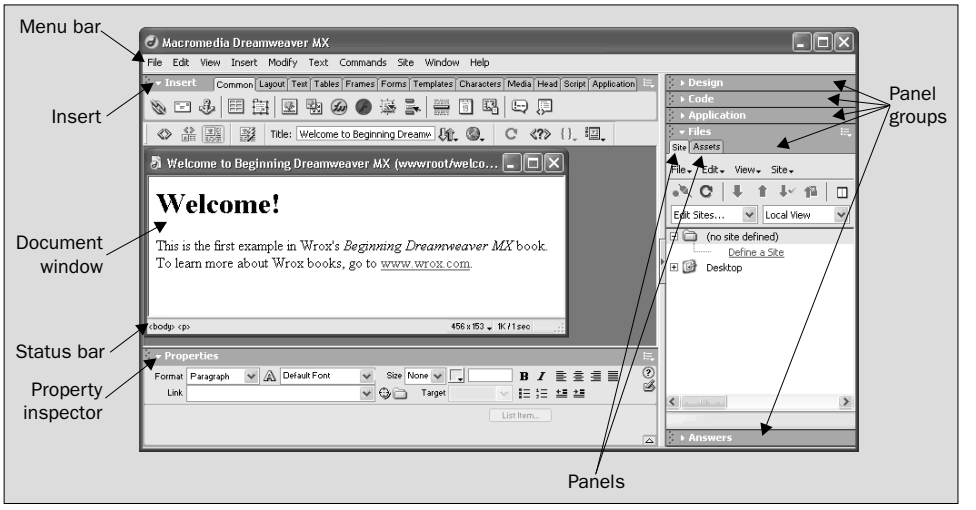
This shows the code that the web server sent to the browser, and which the browser is using to build the page.

In the next chapter, we'll begin to create web pages that make use of client-side dynamic page techniques; and in Section 2 we'll begin to use server-side dynamic techniques using ASP. Let's complete this chapter by taking a quick tour of the Dreamweaver workspace.

The Dreamweaver Workspace

Dreamweaver's **workspace** is standardized with the entire Macromedia MX environment. Many of the features here work the same way as their counterparts in the other Studio MX programs. So, features that you see in Dreamweaver MX also work in a similar fashion in Flash MX, Fireworks MX, and so on. This reduces the overall learning curve for the MX environment, so that once you've got used to Dreamweaver, you'll find it much easier to get started in the other Studio MX programs.

Some of these will seem familiar to you already, while others will be less so:



Note that:

- Dreamweaver's **menu bar** behaves in much the same way as the menu bar in any other Windows-based application, so we will not discuss it here.
- We've seen a little of the **document window** in the example earlier in this chapter – we saw that when we use it to look at an `.htm` file it has a Code View and a Design View, and that they have different but related purposes.
- We'll defer discussion of the status bar for now, but will return to it in Chapter 2.

Let's look at some of the other things that we find here.

Panel Groups and Panels

On the right-hand side of the workspace are the **panel groups**. The following screenshot shows five panel groups (Design, Code, Application, Files, and Answers):

Much the work we will do here will be via the panel groups. Each panel group is (unsurprisingly) just a group of related **panels**. The panels provide features and functionality that helps us develop, configure, and manage our web pages and websites.

In the previous screenshot, the Design panel group has been expanded to reveal that it contains three panels: CSS Styles, HTML Styles, and Behaviors. To expand or collapse a panel group, you simply click on the panel group name, or on the little white **expander arrow** that is immediately to the left of the panel group name. We'll be using these panels over the coming chapters, so we'll defer detailed explanation of their purpose for now.

Hiding and Revealing Panel Groups

You can hide panel groups, or make them visible, to suit your demands. As you get more familiar with Dreamweaver, you may decide to hide some of the panel groups that you rarely use and make visible other panel groups that are not shown by default:

- To hide a panel group, right-click on the panel group and select Close Panel Group
- To pake a panel group visible is slightly more complex, because you need to know which *panel* you want to use. For example, if you want to view the CSS Styles panel, then select Window | CSS Styles from the menu bar (you can access other panels in the same way). When you do that, the panel group containing this panel will appear.

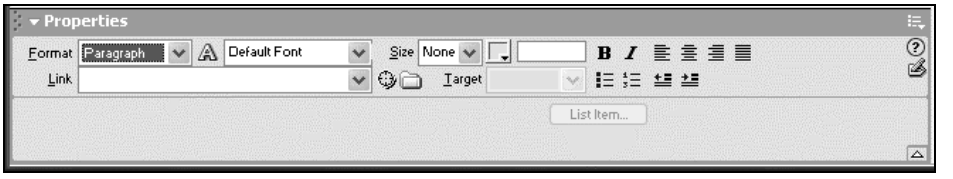
Moving and Docking Panel Groups

To move a panel group around the workspace, you must pick it up by its **gripper** – the pattern of five dots in the top left corner of the panel group – and move it to its new location. In this way, you can:

- **Undock** a panel group – so that it is not displayed with the other panel groups, but floats in a separate "window" that is disconnected from but related to the main Dreamweaver workspace window
- **Dock** an undocked panel group – to move it back into the main Dreamweaver workspace window
- **Change the order** of the docked panel groups – by dragging the panel group by its gripper and dropping it in its new location

The Property Inspector

At the bottom of the workspace is the **property inspector**. The property inspector is the place where we can view and change the properties of whatever element or item you've selected. So, if your cursor is flashing inside an HTML paragraph, then the property inspector shows the properties of that paragraph. If you select a JavaScript behavior, then it shows the properties of that behavior:



This screenshot shows the properties of an HTML paragraph (<p>) element. It shows that the font selected for this paragraph is the default font, and that we have not set a font size, and so on. On the right, there are buttons to enable us to change the font to bold or italic, and to adjust the alignment (to left, center, right, or fully justified), and for adding bullets, and various other things. We'll make good use of these over the coming chapters.

The Insert

Along the top of the screen is a special panel group, called the Insert:

In this panel group we have a set of panels that contain many of the tools commonly used in building a website – for controlling layout and text, and creating and controlling tables, frames, forms, and so on. The Characters panel allows you to insert special characters into your text; the Script panel allows you to control scripts. We will revisit the panels of the Insert panel group extensively over the course of this book.

Setting up the Preview Browsers

The purpose of this book is to build websites, and therefore we're going to need a browser to test the fruits of our work. As we've already seen, Dreamweaver allows us to view a web page by using the *F12* (Preview in Browser) option. In fact, there's more to this than we've discussed, and we can take this a little further now.

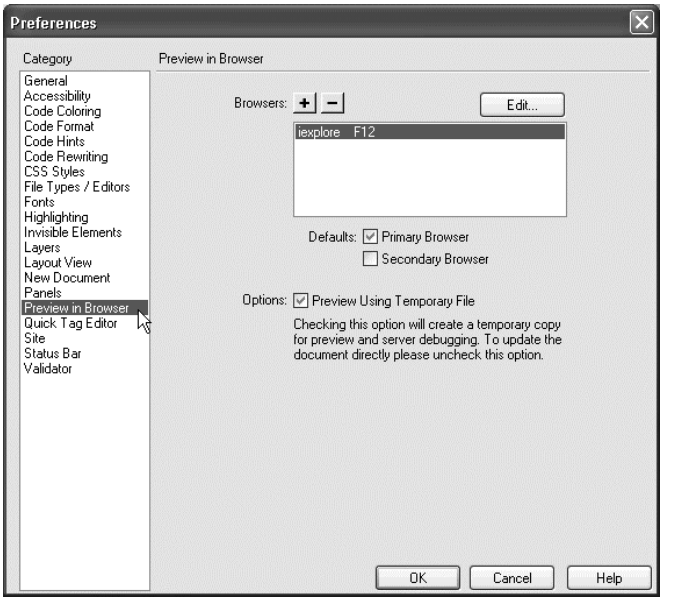
Dreamweaver is probably already configured to allow you to use Microsoft's Internet Explorer browser to view web pages. In fact, it's a good idea to have the latest versions of both Internet Explorer and Netscape Navigator installed, and to configure Dreamweaver to give you an *F12*-type Preview in Browser shortcut for each browser – so that you can easily check what you've done using whichever browser you're interested in.

> *Many web developers also retain earlier versions of these browsers. Remember, when you publish your website on the Internet, there is no guarantee that all of your visitors will be using the latest and greatest browsers. For that reason, it's a good idea to test your work under as many conditions as possible, and with as many different browsers as possible.*
>
> *As we progress through this book, we will see some additional tools that Dreamweaver has to assist us with browser compatibility.*

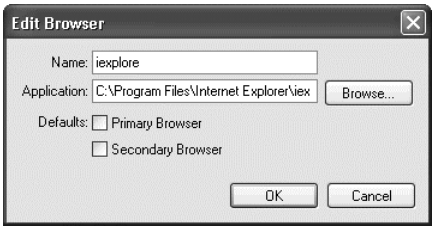
So, for the final exercise in this chapter, we'll set up this very useful bit of configuration. Before you begin this exercise, you might like to see what browsers you have installed on your machine, and perhaps install another one. For example, you can get the latest version of Internet Explorer from http://www.microsoft.com; you can get the most recent Netscape Navigator browser from http://www.netscape.com; and you can get the Opera browser from http://www.opera.com.

Try It Out — Configuring Dreamweaver for Browser-Based Page Previews

1. In Dreamweaver, select Edit | Preferences from the menu bar – this will show the Dreamweaver Preferences dialog. When you're there, select the Preview in Browser category:



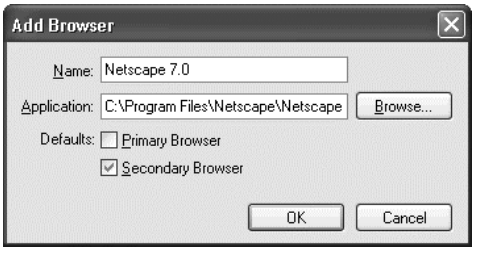
2. First, let's look at the browser for which Dreamweaver has already been configured. If your Browsers box contains an entry, click on it and then click the Edit... button:



This will tell you the name of the browser. It also specifies the file that runs whenever you use this option. (The screenshot above shows what you'll see if your Dreamweaver installation is configured to use Internet Explorer for previewing web pages.) Don't change the settings here; just click OK to close the dialog.

3. Now let's add a second browser. Click on the + button. This will bring up the Add Browser dialog. If your Dreamweaver installation is already configured to use Internet Explorer (as in Step 2), and you've installed Netscape Navigator onto your machine, then you could configure Dreamweaver to allow previews in Netscape Navigator here. To do that:

- For the Name field, type Netscape 7.0 (or something similar)
- For the Application field, browse to the executable file for your Netscape browser (it is probably somewhere like C:\Program Files\Netscape\Netscape\Netscp.exe):

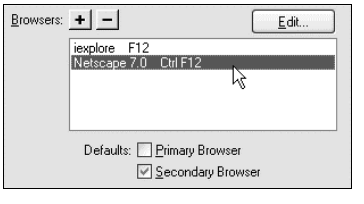


Alternatively, if Dreamweaver is not already configured for Internet Explorer and you want to configure it here, then:

- For the Name field, type Internet Explorer 6.0 (or something similar)
- For the Application field, browse to the executable file for your Internet Explorer browser (it is probably somewhere like C:\Program Files\Internet Explorer\iexplore.exe)

Also you can click the Primary Browser or Secondary Browser checkbox. (This is simply for convenience: in Dreamweaver, you can use the *F12* shortcut for previewing in the primary browser, and the *Ctrl+F12* shortcut for previewing in the secondary browser.) Then click OK to exit the Add Browser dialog.

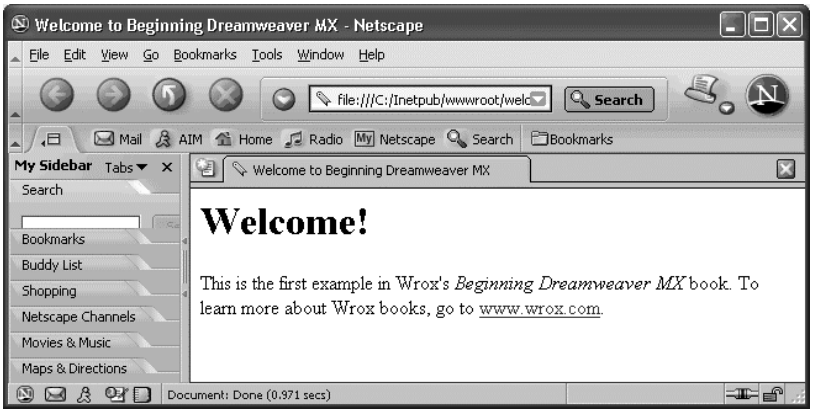
4. In the Preferences dialog, you should now see the browsers listed in the Browsers box:



If you make the Netscape browser your secondary browser, then you should find that the Internet Explorer has been automatically configured to be the primary browser.

5. We have one other thing to decide regarding how Dreamweaver uses browsers, and that is the Preview Using Temporary File option (which is also in the Preferences dialog). Uncheck this checkbox, and then click OK to leave the Preferences dialog.

6. Finally, you can test this configuration. Try clicking *F12* to view welcome.htm in your primary browser. Try clicking *Ctrl+F12* to view it in your secondary browser. Here's what I saw when I viewed welcome.htm in my secondary browser, Netscape Navigator 7.0:



How It Works

You can add as many browsers as you like to the list. You can make one of them your **primary browser**, and another your **secondary browser** – this enables you to preview pages in those browsers using the *F12* or *Ctrl+F12* keyboard shortcuts. If you choose to add other browsers, you can use them by right-clicking the page in the Site panel of the Files panel group and selecting the Preview in Browser... option.

If you leave the Preview Using Temporary File checkbox checked, Dreamweaver will not show you the file you're working on, but will instead use it to create a *temporary* file for the preview. The downside of this is that, if you revise the file, you need to start a new browser session; the refresh will not work. Temporary files can also create a lot of clutter (Dreamweaver is supposed to delete these temporary files when it shuts down, but experience has shown that this does not always happen).

If you uncheck it, you *must* save the file before you can preview it. But this is not really a problem: it's good practice to save your work frequently and before you preview. Also, by doing it this way, you prevent clutter from accumulating and, later on, being uploaded to the server by accident.

Summary

Dreamweaver is a powerful tool for building websites. As we've seen in this chapter, it gives us access to the code, via its Code View; but perhaps more importantly, it also gives us a Design View and a wealth of powerful tools that we can use to generate the code quickly and accurately. This means that we can very quickly get down to the task of building attractive websites with minimal effort.

It's helpful to know what goes on behind the scenes, and to picture the request/response process that occurs whenever a user wants to see a web page. When this happens, the user's browser formulates an HTTP request message (which describes the request), and sends that request across the Internet (or other network) to the web server. When the web server receives the request, it processes it and returns an HTTP response, containing the requested page, to the browser. Then, the browser displays the page on-screen.

This is the request/response in its simplest form, of course – we've discussed the notion of client-side processing and server-side processing, and how they bring dynamic effects and dynamic content to our web pages. We'll begin looking at the former in Chapter 2, and the latter in Section 2.

We're now ready to create a website – The Cooking Place – and to begin building the pages that make up the site, with all the graphics and client-side effects that they involve.

Why Not Try...

1. Take a moment to explore the various panels of the Insert panel group. You'll find that the Insert panel group is a versatile and powerful tool, enabling us to add many different elements and features to our web pages. We'll learn more about them in the coming chapters; for now, why not try using the buttons in the Common panel to add more elements to the .htm page in this using it to add more content to welcome.htm?

2. Take a look at the different panel groups and panels that Dreamweaver provides. The best way to access all the different panels is via the Window menu in the menu bar (this menu bar also tells you the keyboard shortcuts for these panels). We have already seen a little of the Site panel, the Insert panel group, and the Properties panel (the property inspector) – we'll meet many of the others over the course of the coming chapters. For now, it's a good idea to have a look round and see what's available.