PART I

Understanding (X)HTML

✦ ✦ ✦ ✦

In This Part

✦ ✦ ✦ ✦

P1: JYS
WY022-01 WY022/Pfaffenberger WY022/Pfaffenberger-v3.cls June 10, 2004 11:19

CHAPTER 1

Introducing the Web and HTML

✦ ✦ ✦ ✦

In This Chapter

✦ ✦ ✦ ✦

This chapter addresses the questions most people have when they're getting started with HTML/XHTML, such as what is the difference between HTML and XHTML, and when do Cascading Style Sheets (CSS) come into play? If you're already familiar with the basic concepts discussed here, you can get started with practical matters in Chapter 2. Still, I encourage you to at least skim this chapter, making sure you understand the very important distinction between structure and presentation (see *What Is CSS?*) and how HTML, XML, and XHTML are related (see *What Is XHTML?*).

What Is the World Wide Web?

The World Wide Web—the Web, for short—is a network of computers able to exchange text, graphics, and multimedia information via the Internet. By sitting at a computer that is attached to the Web, using either a dialup phone line or a much faster broadband (Ethernet, cable, or DSL connection), you can visit Web-connected computers next door, at a nearby university, or halfway around the world. And you can take full advantage of the resources these computers make available, including text, graphics, videos, sounds, and animation. Think of the Web as the multimedia version of the Internet, and you'll be right on the mark.

How Does the Web Work?

The computers that make all these Web pages available are called *Web servers*. On any computer that's connected to the Web, you can run an application called a Web browser. Technically, a Web browser is called a *Web client*—that is, a program that's able to contact a Web server and request information. When the Web server receives the requested information, it looks for this information within its file system, and sends out the requested information via the Internet.

They all speak a common "language," called HyperText Transfer Protocol (HTTP). (HTTP isn't really a language like the ones people speak. It's a set of rules or procedures, called *protocols,* that enables computers to exchange information over the Web.) Regardless of where these computers reside—China, Norway, or Austin, Texas—they can communicate with each other through HTTP.

The following illustrates how HTTP works (see Figure 1-1):

✦ Most Web pages contain *hyperlinks,* which are specially formatted words or phrases that enable you to access another page on the Web. Although the hyperlink usually doesn't make the address of this page visible, it contains all the information needed for your computer to request a Web page from another computer.

✦ When you click the hyperlink, your computer sends a message called an *HTTP request*. This message says, in effect, "Please send me the Web page that I want."

✦ The Web server receives the request, and looks within its stored files for the Web page you requested. When it finds the Web page, it sends it to your computer, and your Web browser displays it. If the page isn't found, you see an error message, which probably includes the HTTP code for this error: 404, "Not Found."

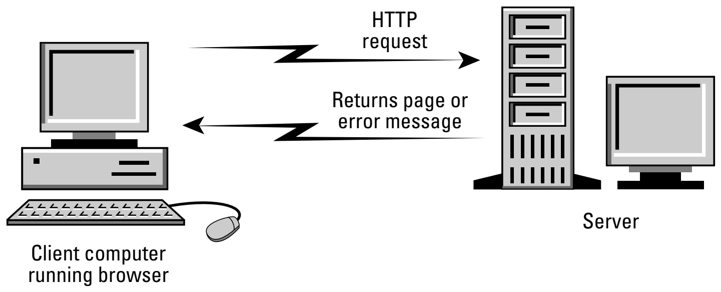


Figure 1-1: The client requests the page. Then the server evaluates the request and serves the page or an error message.

What Is Hypertext?

You probably noticed the word "hypertext" in the spelled-out version of HTTP, Hypertext Transfer Protocol. Originated by computing pioneer Theodore Nelson, the term "hypertext" doesn't mean "text that can't sit still," although some Web authors do use a much-despised HTML code that makes the text blink on-screen. Instead, the term is an analogy to a time-honored (but physically impossible) science fiction concept, the hyperspace jump, which enables a starship to go immediately from one star system to another. Hypertext is a type of text that contains hyperlinks (or just *links* for short), which enable the reader to jump from one hypertext page to another. You may also hear the word *hypermedia.* A hypermedia system works just like hypertext, except that it includes graphics, sounds, videos, and animation as well as text.

In contrast to ordinary text, hypertext gives readers the ability to choose their own path through the material that interests them. A book is designed to be read in sequence: Page 2 follows page 1, and so on. Sure, you can skip around, but books don't provide much help, beyond including an index. Computer-based hypertexts let readers jump around all they want. The computer part is important because it's hard to build a hypertext system out of physical media, such as index cards or pieces of paper.

The Web is a giant computer-based hypermedia system, and you've probably already done lots of jumping around from one page to another on the Web—it's called *surfing.* If one Web page doesn't seem all that interesting once you visit, you can click another link that seems more related to your needs (and so on). The Web makes surfing so easy that you'll need to give some thought to keeping people on your sites—keeping them engaged and interested—so they won't surf away!

Where Does HTML Fit In?

Hypertext Markup Language (HTML) enables you to mark up text so that it can function as hypertext on the Web. The term *markup* comes from printing; editors mark up manuscript pages with funny-looking symbols that tell the printer how to print the page. HTML consists of its own set of funny-looking symbols that tell Web browsers how to display the page. These symbols, called *elements,* include the ones needed to create hyperlinks.

The invention of HTML

HTML and HTTP were both invented by Tim Berners-Lee, who was then working as a computer and networking specialist at a Swiss research institute. He wanted to give the Institute's researchers a simple markup language, which would enable them to share their research papers via the Internet. Berners-Lee based HTML on Standard Generalized Markup Language (SGML), an international standard for marking up text for presentation on a variety of physical devices. The basic idea of SGML is that the document's *structure* should be separated from its *presentation:*

✦ *Structure refers* to the various components or parts of a document that authors create, such as titles, paragraphs, headings, and lists. For example, you're reading an item in an *unordered list,* as it is termed in SGML (most people use the more familiar *bulleted list*). In SGML, you mark up this item as a bulleted list, but you don't say anything about how it's supposed to look. That's left up to whatever device displays or prints the marked-up file.

✦ *Presentation* refers to the way these various components are actually displayed by a given media device, such as a computer or a printer. For example, this book displays this bulleted list item with an indentation and other special formatting.

What's so great about separating structure from presentation? There are several very important advantages:

✦ *Authors usually aren't very good designers.* It's wise, especially in large organizations, to let writers compose their documents, and let designers worry about how the documents are supposed to look. That's particularly true when an organization has a corporate look or style, such as Apple Computer's standard typeface, which you'll see in all of its documents. The designers make sure that every document produced within the organization conforms to that style. So SGML doesn't contain any features that control presentation.

✦ *If markup consists of structure alone, the document's appearance can be changed quickly.* All that's necessary is to change the presentation settings on whatever device is displaying the document.

✦ *Documents containing only structural markup are much easier and cheaper to maintain.* When presentation markup is included along with structural markup, the document becomes an unmanageable mess, and maintenance costs skyrocket.

✦ *If a document contains only structural markup, it is more accessible to people with limited vision or other physical limitations***.** For example, a document marked up structurally might be presented by a Braille printer for those with limited vision, or by a text reader for those with limited hearing.

Sounds great, right? Still, from the beginning, HTML didn't make the structure versus presentation distinction as clearly as SGML purists would have liked. And as HTML developed and the Internet became a commercial network, Web authors demanded more tools to make their documents look attractive on-screen. The companies that make Web browsers responded by introducing new, nonstandardized HTML elements that contained presentation information. By 1996, many Web experts were worried that HTML standards were spiraling out of control. The newly founded World Wide Consortium, hoping to keep at least some kind of standard in place, tried to standardize existing practices, including the use of presentation and structure. The result was the W3C's HTML 3.2 standard, which is still widely used. But organizations found that HTML 3.2 exposed them to excessive maintenance costs. The SGML purists were right: Structure and presentation should have been kept separate.

A short history of HTML

To date, HTML has gone through four major standards, including the latest 4.01. In addition to the HTML standards, Cascading Style Sheets and XML have also provided valuable contributions to Web standards.

The following sections provide a brief overview of the various versions and technologies.

HTML 1.0

HTML 1.0 was never formally specified by the W3C because the W3C came along too late. HTML 1.0 was the original specification Mosaic 1.0 used, and it supported few elements. What you couldn't do on a page is more interesting than what you could do. You couldn't set the background color or background image of the page. There were no tables or frames. You couldn't dictate the font. All inline images had to be GIFs; JPEGs were used for out-of-line images. And there were no forms.

Every page looked pretty much the same: gray background and Times Roman font. Links were indicated in blue until you'd visited them, and then they were red. Because scanners and image-manipulation software weren't as available then, the image limitation wasn't a huge problem. HTML 1.0 was only implemented in Mosaic and Lynx (a text-only browser that runs under UNIX).

HTML 2.0

Huge strides forward were made between HTML 1.0 and HTML 2.0. An HTML 1.1 actually did exist, created by Netscape to support what its first browser could do. Because only Netscape and Mosaic were available at the time (both written under the leadership of Marc Andreesen), browser makers were in the habit of adding their own new features and creating names for HTML elements to use those features.

Between HTML 1.0 and HTML 2.0, the W3C also came into being, under the leadership of Tim Berners-Lee, founder of the Web. HTML 2.0 was a huge improvement over HTML 1.0. Background colors and images could be set. Forms became available with a limited set of fields, but nevertheless, for the first time, visitors to a Web page could submit information. Tables also became possible.

HTML 3.2

Why no 3.0? The W3C couldn't get a specification out in time for agreement by the members. HTML 3.2 was vastly richer than HTML 2.0. It included support for style sheets (CSS level 1). Even though CSS was supported in the 3.2 specification, the browser manufacturers didn't support CSS well enough for a designer to make much use of it. HTML 3.2 expanded the number of attributes that enabled designers to customize the look of a page (exactly the opposite of HTML 4). HTML 3.2 didn't include support for frames, but the browser makers implemented them anyway.

Note

A page with two frames is actually processed like three separate pages within your browser. The outer page is the *frameset.* The frameset indicates to the browser, which pages go where in the browser window. Implementing frames can be tricky, but frames can also be an effective way to implement a Web site. A common use for frames is navigation in the left pane and content in the right.

HTML 4.0

What does HTML 4.0 add? Not so much new elements—although those do exist—as a rethinking of the direction HTML is taking. Up until now, HTML has encouraged interjecting presentation information into the page. HTML 4.0 now clearly deprecates any uses of HTML that relate to forcing a browser to format an element a certain way. All formatting has been moved into the style sheets. With formatting information strewn throughout the pages, HTML 3.2 had reached a point where maintenance was expensive and difficult. This movement of presentation out of the document, once and for all, should facilitate the continued rapid growth of the Web.

Tip

Use the W3C's MarkUp Validation Service, available at `http://validator.w3.org/`, to check your HTML against most of the versions mentioned in this chapter.

XML 1.0

Extensible Markup Language (XML) was originally designed to meet the needs of large-scale electronic publishing. As such, it was designed to help separate structure from presentation and provide enough power and flexibility to be applicable in a variety of publishing applications. In fact, many modern word processing programs contain XML components or even export their documents in XML-compliant formats.

CSS 1.0 and 2.0

Cascading Style Sheets (CSS) were designed to help move formatting out of the HTML specification. Much like styles in a word processing program, CSS provides a mechanism to easily specify and change formatting without changing the underlying code. The "cascade" in the name comes from the fact that the specification allows for multiple style sheets to interact, allowing individual Web documents to be formatted slightly different from their kin (following department document guidelines but still adhering to the company standards, for example). The second version of CSS (2.0) builds on the capabilities of the first version, adding more attributes and properties for a Web designer to draw upon.

HTML 4.01

HTML 4.01 is a minor revision of the HTML 4.0 standard. In addition to fixing errors identified since the inception of 4.0, HTML 4.01 also provides the basis for meanings of XHTML elements and attributes, reducing the size of the XHTML 1.0 specification.

XHTML 1.0

Extensible HyperText Markup Language (XHTML) is the first specification for the HTML and XML cross-breed. XHTML was created to be the next generation of markup languages, infusing the standard of HTML with the extensibility of XML. It was designed to be used in XML-compliant environments, yet compatible with standard HTML 4.01 user agents.

So who makes the rules?

Every organization has its own rule-making body. In the case of the Web, the rule-making body is the World Wide Web Consortium (W3C). The W3C is composed of representatives from a number of high-tech companies who want to have a say in the standards. The W3C tries to balance the interests of the academy, the companies

P1: JYS
WY022-01 WY022/Pfaffenberger WY022/Pfaffenberger-v3.cls June 10, 2004 11:19

producing the Web browsers (notably Netscape and Microsoft), and the technology. The W3C pulls together committees with representatives from interested members and puts the specifications in writing for HTTP and HTML, as well as a host of additional Web standards, including CSS. If the W3C weren't maintaining all these standards, the Web wouldn't be as easy to use; in fact, it might not have become anywhere near as popular as it is. You can visit their Web site at `http://www.w3c.org`.

Buzz and scrambling

How does the W3C decide when a new technology must be standardized or a new version of an existing technology must be developed? Newsgroups and mailing lists exist where leading figures in the relevant field talk about the shortcomings of an existing version or the idea of a new technology (that's the buzz). If a ground swell of support seems to exist for a new technology or a new version, the W3C begins the process of specifying it.

Something else, however, carries more weight and more urgency than discussion by agitators and activists. This is ongoing development by software developers (that's the scrambling). In reality, the W3C is mostly involved in trying to standardize the proprietary extensions developed by software developers, such as Netscape and Microsoft. If the W3C didn't do this, within two versions of their browsers, HTML might not run the same (or at all) on both systems. The W3C reins them in to some degree. Neither wants to produce a browser that lacks support for recommended HTML elements, so even if Netscape introduced a new element, Microsoft will incorporate that element in the subsequent version of their own browser—after an official recommendation by the W3C (and vice versa).

Committees and working drafts

When a new technology or a new version of an existing technology is required, the W3C convenes a committee of interested parties to write the specification. The committee publishes its work on an ongoing basis as a working draft. The point of publishing these working drafts is this: Software developers who want to implement the new technology or the new features of the new version can get a jump on things and build their product to incorporate the new features. When the specification is finalized and developers are ready to use it, products that implement it are on the market.

There is also the issue of books. You want books on new technologies to be in the bookstores the day the recommendation is finalized. For this to happen, authors must write the books using the working drafts—a moving target—as the reference materials. Working drafts have changed during the writing of this book. Sometimes this works and sometimes it doesn't. If the specification changes radically from the working draft to the final version, the book will be inaccurate.

Voting process

Democracy: You just can't get away from it. When a working draft reaches a point where the committee is pleased and believes it is complete, the working draft is released to the public as a proposed recommendation. Members of the W3C have up to six weeks to vote on it—votes can take the form of any one of three choices: yes, yes if certain changes are made, or no. At the conclusion of the voting process, the W3C can recommend the specification officially, make the requested changes and recommend the specification with the changes, or discard the proposal.

What Is CSS?

In 1997, the World Wide Web Consortium released the first HTML 4 recommendation, the first to embody a serious effort to separate structure from presentation. The W3C envisioned a transitional period, in which Web authors would continue to use some presentation features in their pages, but the end point was clear: Any Web page that wanted to conform strictly to HTML would have to omit presentation-related coding.

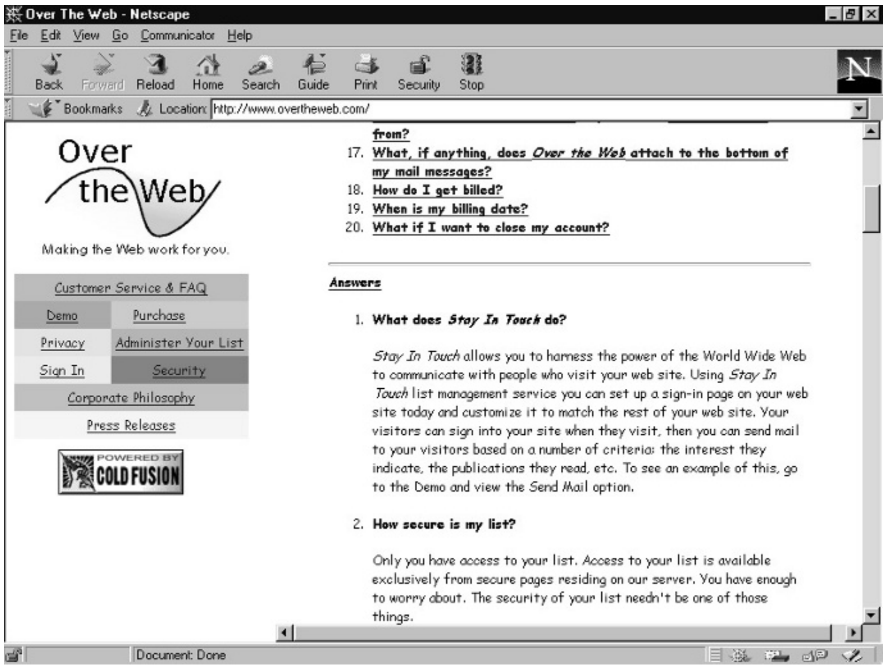
To see for yourself how difficult maintaining HTML 3.2 code can be, consider the following HTML:

```
<li><FONT SIZE="+1" FACE="comic sans ms" FAMILY="sans-serif"
COLOR="#0000FF"><P><A name="do"></a><B>What does <i>Stay In
Touch</i> do?</B></P></FONT>
<FONT SIZE="-1" FACE="comic sans ms" FAMILY="sans-serif"
COLOR="#000000"><P><i>Stay In Touch</i> allows you to harness
the power of the World Wide Web to communicate with people
who visit your web site. Using <i>Stay In Touch</i> list
management service you can set up a sign-in page on your web
site today and customize it to match the rest of your web
site. Your visitors can sign into your site when they visit,
then you can send mail to your visitors based on a number of
criteria: the interest they indicate, the publications they
read, etc. To see an example of this, go to the Demo and view
the Send Mail option.</P></FONT>
<li><FONT SIZE="+1" FACE="comic sans ms" FAMILY="sans-serif"
COLOR="#0000FF"><P><A name="security"></a><B>How secure is my
list?</B></P></FONT>
<FONT SIZE="-1" FACE="comic sans ms" FAMILY="Sans Serif"
COLOR="#000000"><P>Only you have access to your list. Access
to your list is available exclusively from secure pages
residing on our server. You have enough to worry about. The
security of your list needn't be one of those
things.</P></FONT>
```

Figure 1-2 shows what this HTML code looks like in a full page on a PC, while Figure 1-3 shows what that same page looks like on a Mac (notice that the font is slightly different).

The maintenance nightmare

From looking at the HTML and then seeing the HTML interpreted by the browser, you can pretty much tell what part of the text is instructions to the browser and


Over The Web - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://www.overtheweb.com/
Over the Web
Making the Web work for you.
Customer Service & FAQ
Demo Purchase
Privacy Administer Your List
Sign In Security
Corporate Philosophy
Press Releases
POWERED BY COLD FUSION
Document: Done

from?
17. What, if anything, does *Over the Web* attach to the bottom of my mail messages?
18. How do I get billed?
19. When is my billing date?
20. What if I want to close my account?

Answers

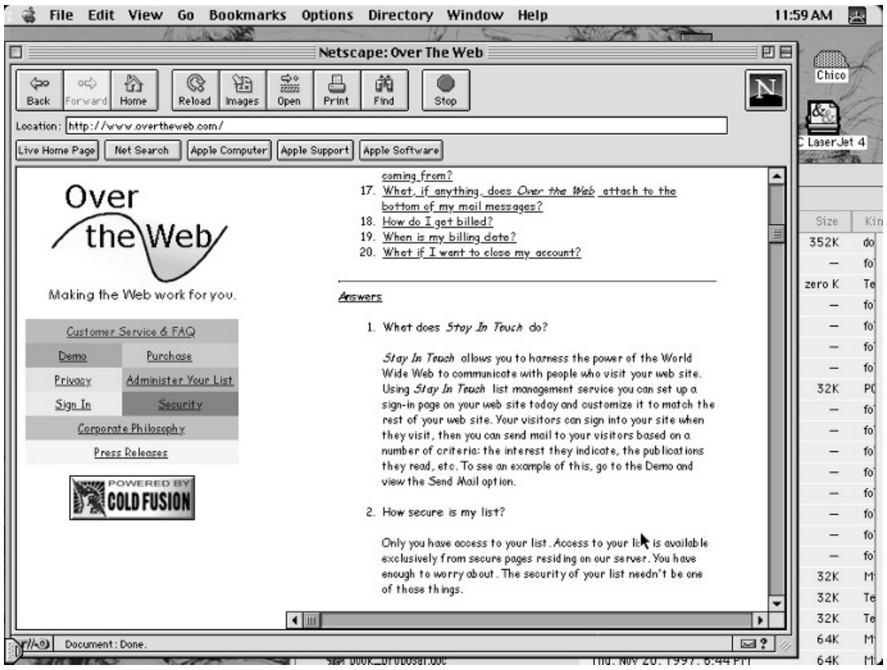
1. What does *Stay In Touch* do?

Stay In Touch allows you to harness the power of the World Wide Web to communicate with people who visit your web site. Using *Stay In Touch* list management service you can set up a sign-in page on your web site today and customize it to match the rest of your web site. Your visitors can sign into your site when they visit, then you can send mail to your visitors based on a number of criteria: the interest they indicate, the publications they read, etc. To see an example of this, go to the Demo and view the Send Mail option.

2. How secure is my list?

Only you have access to your list. Access to your list is available exclusively from secure pages residing on our server. You have enough to worry about. The security of your list needn't be one of those things.

Figure 1-2: How a PC browser displays the HTML code.


File Edit View Go Bookmarks Options Directory Window Help 11:59 AM
Netscape: Over The Web
Back Forward Home Reload Images Open Print Find Stop
Location: http://www.overtheweb.com/
Live Home Page Net Search Apple Computer Apple Support Apple Software
Chico
LaserJet 4
Over the Web
Making the Web work for you.
Customer Service & FAQ
Demo Purchase
Privacy Administer Your List
Sign In Security
Corporate Philosophy
Press Releases
POWERED BY COLD FUSION
Document: Done.

coming from?
17. What, if anything, does *Over the Web* attach to the bottom of my mail messages?
18. How do I get billed?
19. When is my billing date?
20. What if I want to close my account?

Answers

1. What does *Stay In Touch* do?

Stay In Touch allows you to harness the power of the World Wide Web to communicate with people who visit your web site. Using *Stay In Touch* list management service you can set up a sign-in page on your web site today and customize it to match the rest of your web site. Your visitors can sign into your site when they visit, then you can send mail to your visitors based on a number of criteria: the interest they indicate, the publications they read, etc. To see an example of this, go to the Demo and view the Send Mail option.

2. How secure is my list?

Only you have access to your list. Access to your list is available exclusively from secure pages residing on our server. You have enough to worry about. The security of your list needn't be one of those things.

Figure 1-3: The previous text displayed in a browser on a Mac.

what part is the content. But would you feel comfortable making changes to the content—say, adding another bulleted set of questions and answers? Probably not. With all those codes embedded within the text, you might mess something up. And you probably wouldn't want someone else who didn't know what all those codes meant doing it either.

The worst maintenance nightmares occur when you want to change the look of your pages throughout your Web site. Because the presentation code has to be included in every page, you have to change every page to change the look of your site.

Consider the site map shown in Figure 1-4. Every screen should have the same formatting: same font, same heading sizes, same alignment, same text color, and same background color. With HTML 3.2, you could do this only by inserting all the needed presentation code on every single page.

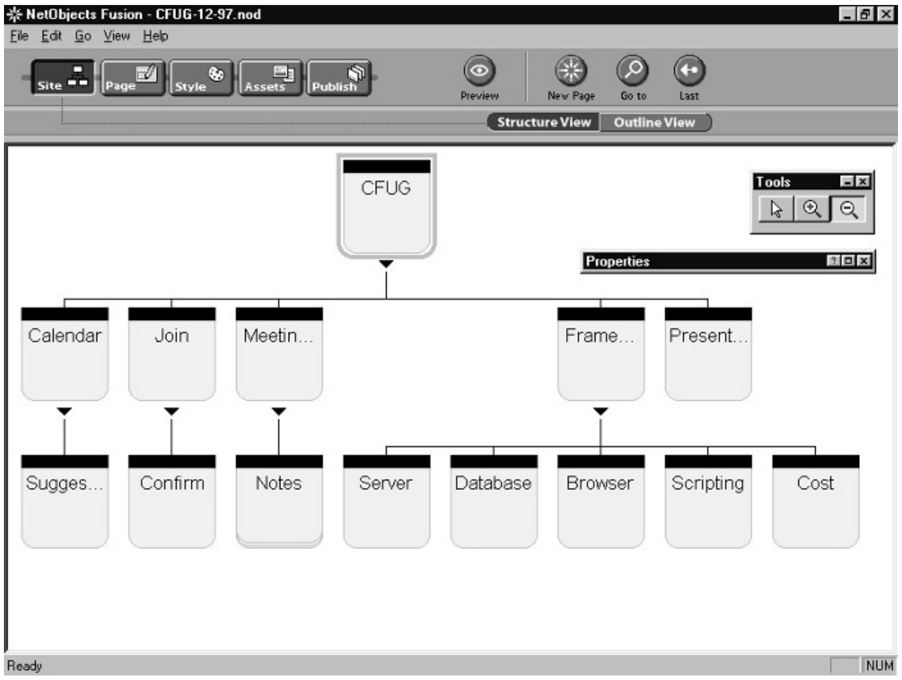


Figure 1-4: Map of a Web site.

HTML 4.01 enables you to return to the ideal of separating structure and presentation. What does this mean to you and your ability to maintain a site? It's simple. HTML that contains nothing but structural code is vastly simpler and cheaper to maintain.

Consider the following code. It produces the same results as the previous example in the browser. Notice there is no formatting. All the HTML you see is related to the structure.

```
<li>What does <i>Stay In Touch</i> do?</li>
<p><i>Stay In Touch</i> allows you to harness the power of
the World Wide Web to communicate with people who visit your
web site. Using <i>Stay In Touch</i> list management service,
you can set up a sign-in page on your web site today and
customize it to match the rest of your web site. Your
visitors can sign into your site when they visit. Then you
can send mail to your visitors based on a number of criteria:
the interest they indicate, the publications they read, etc.
To see an example of this, go to the Demo and view the Send
Mail option.</p>
<li>How secure is my list?</li>
<p>Only you have access to your list. Access to your list is
available exclusively from secure pages residing on our
server. You have enough to worry about. The security of your
list needn't be one of those things.</p>
```

Note

The use of the italic tags (`<i>`) in the preceding code is arguably "formatting" and is used to simplify the example while conforming to the visual style of the text "*Stay In Touch.*" When using HTML 4.01 and CSS it might be better to use span tags (`<span>`) to refer to a CSS class instead of directly specifying the italic text attribute. See Chapter 16 for more information on styles and span tags.

How comfortable would you be updating the previous HTML? How about if you needed to add another set of questions and answers? Already, you can see that using HTML 4.01 makes a world of difference.

There's only one problem. The simpler, HTML 4.01-compliant code looks just terrible on-screen; with no presentation information in the code, the browser falls back on its default presentation settings.

Enter CSS

By themselves, strictly conformant HTML 4 documents are ugly. Web authors would never have accepted HTML 4 if the W3C had not also developed Cascading Style Sheets (CSS). In brief, CSS enables Web authors to specify presentation information without violating the structure versus presentation distinction. The information the browser must know to format the previous text is stored separately, in a *style sheet*. The style sheet lists the presentation styles that the browser should use to display the various components of the document, such as headings, lists, and paragraphs. The style sheet is kept separate from the marked-up text. It can be stored in a special section of the HTML document itself, away from the document's text, or in a separate file entirely.

The idea and the name come from the publishing industry, where style sheets are still used today. And they're cutting costs wherever Web documents are created and maintained.

Think back to the problem of updating a Web site's look, discussed earlier. Without CSS, you'd have to make changes to the presentation code in each and every page.

Thanks to CSS, all you have to do is make changes to the single, underlying style sheet that every page uses, and the entire site's appearance changes.

What does "cascading" mean?

In Cascading Style Sheets, the term "cascading" refers to what computer people call the *order of precedence*—that is, which style information comes first in the style pecking order. Here's the order:

✦ *Element-specific* style information comes first. This is style information that's embedded within the HTML. But wait—doesn't this violate the structure versus presentation rule? Yes, but sometimes it's necessary. If element-specific information is given, it takes precedence over page-specific and global styles.

✦ *Page-specific* style information is kept in a special section of the document, called the *head,* that's separate from the text. It defines the way a particular page looks. If page-specific information is given, it takes precedence over global styles.

✦ *Global styles* are specified in a separate style sheet file. They come into play unless conflicting element- or page-specific styles are given.

See Figure 1-5 for a summary of these points.

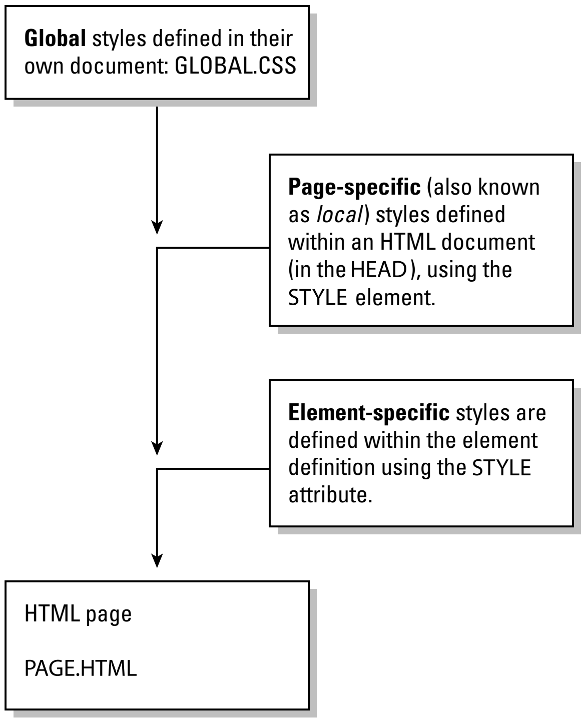

Global styles defined in their own document: GLOBAL.CSS
Page-specific (also known as *local*) styles defined within an HTML document (in the HEAD), using the STYLE element.
Element-specific styles are defined within the element definition using the STYLE attribute.
HTML page
PAGE.HTML

Figure 1-5: The cascading model of style definitions.

HTML 4.01 almost eliminates the need to have an HTML expert perform site maintenance. This means HTML 4.01 helps reduce the cost of maintaining your Web site. When was the last time you heard anything about reducing costs being associated with the Web?

What Is XHTML?

Combined with CSS, HTML 4.0 was a major advance, so one might expect even better versions of HTML in the future, right? Not according to the World Wide Web Consortium. Apart from a minor update (HTML 4.01) in 1999, HTML 4.0 is the last version of HTML. That's because it has been replaced by XHTML, which is the version of HTML you're going to learn in this book.

Actually, there's very little difference between HTML and XHTML. It's a matter of making a few changes to your HTML 4.0 code to make sure it's XHTML-conformant. But there's a much deeper reason for this change. To understand why HTML has become XHTML, you must learn a little about XML, another World Wide Consortium standard.

As you've learned, HTML is based on SGML, an international standard for markup languages. Actually, SGML isn't a markup language in itself. It's a language that's useful for *creating* markup languages. You can use it to make up codes for just about anything you want. For example, an accounting firm could use SGML to mark up the structure of accounting documents; one code could be used to mark daily totals, while a different code could be used to mark monthly totals. To keep a record of all these newly created codes, as well as to specify them for presentation devices, a special file, called a *document type definition (DTD),* is used. HTML 4.01 is defined in a document type definition, written in SGML.

SGML isn't equally loved by all. To many, SGML is outmoded, overly complex, and too difficult to learn. So the World Wide Consortium decided to create a new version of SGML that would be simpler and easier to learn. The result is the Extensible Markup Language (XML). Like SGML, XML enables people to define new markup languages that are exactly suited to their purposes.

Now that you know a little about what XML is, you're ready to understand what XHTML is. Just as HTML is a markup language defined in SGML, XHTML is a markup language defined in XML.

Creating an HTML Document

Creating an HTML document is relatively easy. One of the nice properties of HTML is it is just text. The content is text and the tags are text. As a result, you can write your HTML in any text editor. If you are running any variety of Windows, you can use Notepad, which comes installed with Windows. If you have a Mac on your desk, you can use SimpleText. If you work in UNIX, you can use emacs, vi, jove, pico, or whatever you normally use to edit text. Essentially, any text editor or editor capable of producing text-only documents can be used to create HTML documents.

Writing HTML

What else do you need to know to write your HTML? Presumably, by now, you know the following:

✦ What your purpose is (at least generally)

✦ You need to write your content from your focused message

✦ You mark up your content with HTML tags

✦ You can write your page with a text editor that is already installed on your computer

Obviously, you need to know the elements. But before discussing those, here are a few guidelines about how you should and shouldn't use HTML.

Name your files with a Web-friendly extension

When saving an HTML file you should always give it a `.html` or `.htm` extension. (The former, `.html`, is generally preferred.) This correctly identifies the file as having HTML content so that Web browsers and servers correctly handle it.

Other Web files have their own extensions—for example, most PHP files use `.php`, graphic files use `.jpg`, `.gif`, or `.png`, and so on. This book suggests appropriate extension(s) as each technology is discussed.

Format your text

If you are already writing HTML pages, you may need to break your bad habits. You probably already think in terms of getting the browser to make your page look the right way. And you use HTML to make it do this. You may even use goofy conventions such as 1-pixel-wide clear image files (usually GIFs) and stretch them to indent your paragraphs.

With HTML 4, you needn't out-maneuver the browser. Browsers that support the HTML 4 standards display your pages as you define them—no more of that arrogant printer stuff! And fortunately, with HTML 4, you can define the way you want your pages to look outside of the content, so your HTML won't be all cluttered with tags.

Structure your document

So, if you are not supposed to use HTML to format your pages, how should you use HTML?

HTML defines your document's structure. Then, outside the main body of the document (or even in a separate file, if you prefer), you define the appearance of each element of the structure (just like the publisher and the printer in the previous example).

With few exceptions, you want all your paragraphs to be formatted the same—uniform margins, indents, fonts, spacing between lines, and color.

So, within the main body of your document, you type your text for each paragraph and mark up your document to indicate where each paragraph begins and ends. Then, in a separate location and only once, you define how you want all your paragraphs to look. Existing ways to override this universal definition are discussed later.

The most important concept to remember—and this is a big change for you if you've already been writing HTML 3.2 or earlier versions—is that the HTML only defines the *structure* of your document. The *formatting* of your document is handled separately.

What is so great about this? First, your text doesn't get all cluttered up with tags. And second, you can define the look for your entire site in one place. You simply have every page in the site (even if some pages in your site are being written by people you have never met) point to the style sheet (the place where you put all those style definitions).

Don't I Need a Web Server?

Later chapters will show you how to upload your documents to a dedicated and public Web server where others can see them. In the meantime, you can simply use your computer's hard drive and a local browser to dabble privately with HTML.

Note Server-side technologies such as PHP require an actual Web server.

Simply put the document on your hard drive and direct your browser at the file. For example, in Windows you can double-click an HTML file in Windows Explorer (or any other file manager) to open it in the default browser (normally Internet Explorer). Most browsers also have an Open File option under their main File menu.

Create additional files, directories, and subdirectories on your local hard drive as needed to hold additional pages or levels of a site.

Tip Apache, the Web's most popular HTTP server, is available for several architectures and best of all, it's free to use. If you need a local Web server for testing purposes, visit the Apache Web site (`http://httpd.apache.org/`) for more information or to download a copy for your machine. For more information on Apache, see Wiley's *Apache Server 2 Bible, 2nd Edition*, by Mohammed J. Kabir.

P1: JYS
WY022-01 WY022/Pfaffenberger WY022/Pfaffenberger-v3.cls June 10, 2004 11:19

Summary

This chapter covered the basics of HTML and the Web. Before actually creating Web documents, it is important to understand the evolution of the technologies behind the Web, and the direction they will take in the future. The next few chapters discuss the basic elements of HTML documents and get you on your way to creating your own Web content.

✦ ✦ ✦