CHAPTER 5

# Introducing HTML

**In This Chapter**

What's your Web site's purpose?

Understanding your tools

SGML: The father of HTML

The HTML DTD of SGML

The standardization process

A brief history of HTML, Versions 1 to 4

Writing HTML: A few basics

Understanding the standardization process

Players in the standardization process

The extensions game

So you want to publish a Web site. Why? Unless you have a clear purpose, you won't be able to create a compelling site. Who is your audience? Surely you must have some idea of who cares about what you have to say (other than your mother, and she's just being polite). What do you want to tell people? Your message must be crystal clear. No fancy design will compensate for a murky message. What do you suppose your audience wants to know? Don't assume what you want to tell them and what they want to know are the same. Because you probably can't influence what they want to know, you may have to modify what you want to tell them to answer their needs.

This book teaches HTML 4, but, as mentioned earlier, not everyone who comes to your Web page will be running a browser that supports HTML 4 features. The browsers your visitors use affect how they see your site. In this chapter, you also get a brief overview of the process involved in getting a version of HTML recommended by the World Wide Web Consortium (W3C). You also see examples of the document type definitions that SGML uses to specify HTML. This chapter introduces HTML by taking you through the evolution of HTML from Version 1.0, which was never even published by the W3C, to Version 4.01, the current version.

Standardization is the name of the game on the Web. Because everyone agrees that Web pages should be loaded using the HTTP protocol, every browser can load every standard Web page. When people or companies stop conforming to the standard or, more frequently, develop new standards in areas that aren't yet well developed, standardization becomes a problem. This chapter sorts out the thorny history of standardization. You learn why much of what HTML has to offer (JavaScript, the DOM, Java implementation, and CSS implementation) isn't yet standardized between the major browsers. You also learn a bit about the process that exists within the W3C for creating standards.

# What's Your Purpose?

Think of the last time you saw a presidential press conference or speech. Do you suppose the speechwriter planned ahead which of the president's quotes would make it onto the evening news? You bet the writer did! Did you notice after the speech how all the high mucky-mucks in the president's party had exactly the same thing to say? Sure, they used different words, but they all had the same message. Of course, this is no accident.

You have to plan your own Web site with the same single-mindedness. What is the purpose of putting up your Web site? First, will your site be a transit point or a destination?

## Transit point

A *transit point*, like a train station, is a place people pass through to get where they really want to go. A search engine is a transit point. A list of places to find 100-percent cotton, organic baby clothes is a transit point. Lots of people put up pages full of pointers to other pages. Whether they realize it or not, these are transit points.

Want to put up a commercial Web site? You can make money from advertising banners, if you can attract a crowd to your transit-point Web site.

How long will you stay at a page that does nothing more than point you to other pages? What if you want people to stay in your site and look around, read all the pages, and perhaps contribute their opinions or purchase something? Then you don't want to put up a transit point site; you want to put up a destination site.

## Destination

A destination Web site is the site where you spend time looking for something. You normally won't leave a destination Web site until you have found it or you've given up trying. Then you may take one of the links off one of the pages and see what you can find there.

Five types of destination Web sites exist:

- **Educational.** Teach your visitors about something.
- **Motivational.** Motivate your visitors to take some action.
- **Informational.** Tell your visitors something.
- **Persuasive.** Convince your visitors of something.
- **Sales.** Sell your visitors a product or service.

Frequently, elements of more than one of these exist in a site.

## Educational

Once upon a time, the Web was primarily populated with educational sites. Because the Web's first inhabitants were primarily academics and researchers, this only made sense. Although times have changed, you can still find excellent educational resources on the Web. Figure 5-1 is an excellent example of an education site on the Web.

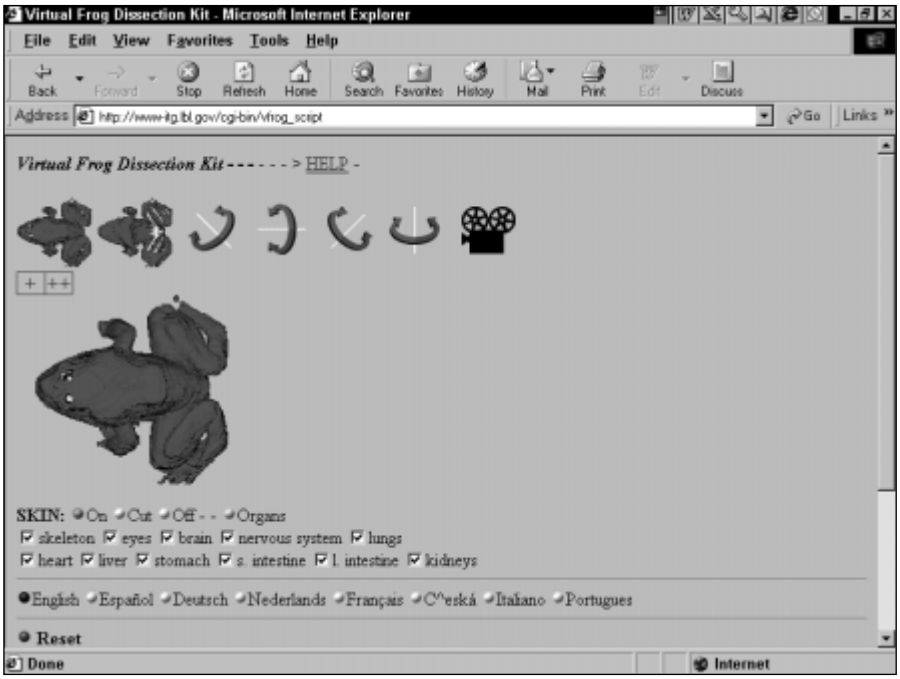


**Figure 5-1:** A frog dissection Web site

## Motivational

Perhaps you are putting up a Web site to get people to take action: give money to their favorite charity, volunteer their time with their local library, call their member of Congress to demand stricter enforcement of immigration laws, or start exercising regularly. You may or may not be able to assume your visitors already believe in the value of your cause. You may have to do some persuading, as well.

Motivational sites are what you probably have in mind if your message is related to improving your community, the society, or the environment. Giving people information (education) is not enough. You probably also want them to take action: join the PTA, drop off food at the soup kitchen, or recycle.

You may not need to get your visitors to call the president, but you may want to assist them in managing their time so they can find time to exercise. Or, you may list local public health clinics in your state where they can get their children vaccinated. If you are going to motivate your visitors to do something, you must give them all the information they currently need to take action.

## Informational

In an informational site, you simply want to tell your visitor something. It could be your family history, the cute tricks your cat does, or the services provided by the local United Way. If your page is strictly informational—such as design tips for quilting (see Figure 5-2)—then you are under less pressure to conform to standards. However, you still want to have a clear message, good content, and a clean design, as this page has.


**Figure 5-2:** Quilting design basics—a lovely example of an informational page

## Persuasive

With a persuasive site or page, you have a clear message. You want your visitor to agree with you about something. Your job is to explain your cause and then support it with enough evidence to win your visitor over to your way of thinking.

If you don't have a clear idea of what you want your visitors to take away from your site — what you want them to remember — then they won't either.

## Sales

This is perhaps the most straightforward purpose of a Web page: selling. In other words, you want your visitor to buy a product or service. What does it take to convince your visitor to make a purchase? You have to educate, inform, persuade, motivate, and finally, sell! On a sales site, you probably want to have some sort of server processor to handle payments and security to insure that information for payments is safe. Figure 5-3 shows a form used on the Anvil Bikes site to capture a prospective buyer's information.


**Figure 5-3:** Most sales sites should culminate in a form that requests basic purchasing information.

# Understanding Your Tools

Have you clarified your message at all since the beginning of this chapter? What is the purpose of your site or page? What kind of site or page will it be? Will it have elements of more than one type of site? Fortunately, you have a few more topics to cover before you begin writing.

As you probably guessed from the title of this book, the primary tool you use to create your Web site is HTML 4. So what is HTML anyway?

Simply put, HyperText Markup Language is a set of rules for marking up text so a browser knows what to do with that text. Do you want to see some HTML right now? While browsing any page on the Web, select View Source or View Document Source from the View menu at the top of your screen. Just in case you're not currently online, here is some HTML from the IDG Books Worldwide Web site.

```
<!-- Hot and Happening -->        <font size="4"><b><tt>Hot and
Happening</tt></b></font><br>
<!-- Help your Valentine get more out of Windows 2000 with
Brian Livingston's <a href="/cgi/fill_out_template.pl?idgbook:
0-7645-3413-0:book-idg::~uid~"> Windows&reg; 2000 Secrets&reg;.
-->        <p><!-- Windows 2000 Trade Show -->
<a href="http://www.idgbooks.com/bookstore/win2000.html"><font
size="4"><b><tt>Windows 2000 is Here</tt></b></font></a><font
size="4"><b><tt> </tt></b></font><br>
```

As you can see, HTML looks like text with other things inserted between less than (<) and greater than (>) signs. If you try to read the previous HTML, you should be able to extract the following text from it: "Hot and Happening Help your Valentine get more out of Windows 2000 with Brian Livingston's Windows 2000 Trade Show."

Your browser makes sense of the tags within the < and > (often referred to collectively as angle brackets) so the text between the codes looks exactly as the designer wants it to look. What a great idea! You write the content, and then you mark up the text to look the way you want using HTML.

## SGML

Standard Generalized Markup Language (SGML) is the basis for HTML. Actually, HTML is a subset of SGML. What you probably didn't realize is that HTML is actually specified in SGML. In other words, SGML is used to define HTML. SGML uses document type definitions (DTDs) to specify itself, the elements of HTML, and XML. (DTDs are covered in detail in Chapter 6.)

SGML became standard in 1988, when it was approved by the International Standards Organization (ISO). Why should you care? You will better understand where HTML is going if you understand where it came from. Briefly then, SGML is a standard for electronic document exchange. It was, in fact, used in the publication of this book!

How does SGML work? When you read a book manuscript, each element on the page (paragraph, section heading, block quote) has its own typographical convention. For example, in this book, anything representing HTML code appears in courier font, such as the previous HTML code. The publisher communicates to the printer that the code sample should be typeset a certain way by marking up the text with special codes and then telling the printer that whenever he sees those codes, he should

change the font to the appropriate font for that element (the publisher gets to define this).

If you've ever tried to write your own Web page, you've noticed you can't always get it to look precisely the way you want. Why is that? You are like the publisher: You mark up your text and tell the printer how you want it to look. The browser is like an arrogant printer, who reads what you have to say and, sometimes, does it his own way despite your instructions.

As a Web-page designer, you are at the mercy of the browser. If a publisher doesn't like the way the printer follows directions, he can take his business elsewhere. If you don't like the way a particular browser handles your Web page, you can't do much about it. Fortunately, this is changing with HTML 4.

### HTML

When the first graphics-capable browser (Mosaic 1.0) was released in 1993, it handled only a small subset of all the SGML tags. Essentially, as a designer, you could center text, choose from one of four font sizes, separate paragraphs with a blank white line, and do a few other things to make your text look nice.

We've come a long way since then. Today you have extensive control over white space and formatting. However, HTML is still only a small subset of SGML. HTML is definitely growing in the direction of SGML.

## The HTML Standardization Process

But how are changes made to the HTML standard? Who controls this gradual movement of HTML towards SGML? The standards-setting body for the World Wide Web is the W3C. Its membership is derived from companies, not individuals. The W3C has a process for recommending versions of HTML and other Web-related technologies. Currently in the hopper are CSS level 3, XML, PICS, and the Web Accessibility Initiative (WAI).

### 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). Nothing gets the W3C activated quite so quickly. 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 one 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 new product to incorporate the new features. When the specification is finalized and developers are ready to use it, products are on the market that implement it.

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, then the book will be inaccurate. Most of the time, working drafts only get thicker as they approach approval time.

## 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.

### 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 as they are today, 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 the HTML 4 Way). HTML 3.2 didn't include support for frames, but the browser makers implemented them anyway.

**Definition**

Frames. 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.


Use the Web Page Purifier, available at `www.delorie.com/web/purify.html`, to check your HTML against most of the versions mentioned in this chapter.

Enough about the history of HTML.

## HTML editors

Where do you write your HTML? 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 flavor of Windows, you can use Notepad, which comes installed with Windows. If you have a Mac on your desk, you can use SimpleText, as shown in Figure 5-4. If you work in UNIX, you can use emacs, vi, jove, pico, or whatever you normally use to edit text.

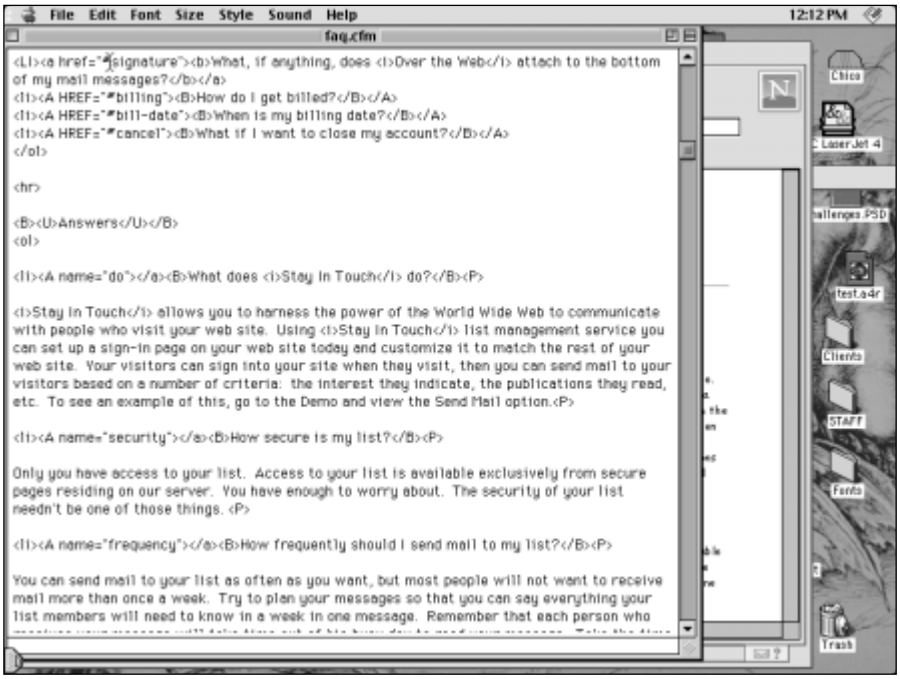


**Figure 5-4:** SimpleText editor


For your first page, we use your regular text editor. Chapter 8 is dedicated to HTML-editing programs that increase your productivity over a plain old text editor. But for now, we focus on the fundamentals: HTML elements and content.

# Writing HTML

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

- 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.

HTML shouldn't be used to format your text: It should be used to structure your document.

## Desperate men and women

As we discussed in Chapter 2, the lengths to which Web designers have gone to format their pages the way they wanted was quite absurd. HTML had a limited tag set, meaning everything you may have wanted to do to your text wasn't necessarily possible. Yet visitors' expectations for graphically appealing pages were still high. Web designers, before HTML 4 came along, had to get creative. One devious device to force indentation of paragraphs or a set amount of white space between paragraphs was the clear GIF. A *GIF* is an image file (called *somename.GIF*). It is possible (in a software package designed for image creation, such as Photoshop) to create a 1×1-pixel image. A *pixel* is the smallest unit of measurement on the screen. Then with HTML, the image can be stretched to fill the desired space. The evil tag would look something like this:

```
<img src="clear.GIF" width="18" height="1">
```

Then the Web designer would write the text for the paragraph. The result would be 18 pixels of white space (or about ¼ inch on most monitors) before the first word of that paragraph. You can imagine what this did to the readability of the HTML when all that junk went at the beginning of the paragraph!

## 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. If you are really underhanded, 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? Glad you asked.

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. Ways exist to override this universal definition, but we discuss them 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 whole 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).

# Including Multimedia

Throughout this explanation, we have been talking about text. What if you want to include images? Sound? Video? Animations? 3D models? HTML 4 supports the use of multimedia in your pages. Chapters 40 and 41 go into detail on adding sound and video to your site.

Two ways exist to include multimedia features in your page: inline and out-of-line, and uses exist for both. The only way to insure your visitor sees or hears your multimedia, however, is to put it inline. For example, advertising banners wouldn't get seen much if they weren't inline!

## Inline

Visitors to your page can see or hear an inline element without taking any additional action. Inline elements are supported by HTML directly. Examples of inline elements are images, sound, and animated images (called *animated GIFs*, because this is the only type of image that supports animation). Figure 5-5 is an example of an inline image.

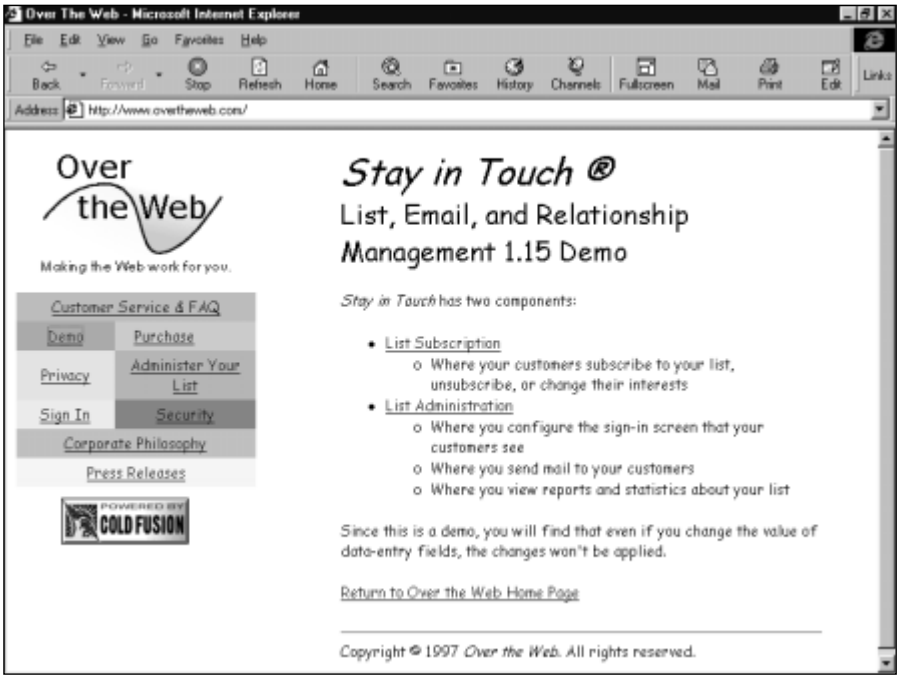


**Figure 5-5:** The Over the Web logo in the upper left-hand corner is an inline image.

Unless an image is huge (and justifiably so), meaning greater than the size of your screen, you probably want to keep your images inline. Exceptions always exist. Say you want to show a map of Chicago. You may allow your visitors to click a neighborhood and see a more detailed map of only that neighborhood. In this case, you need both inline and out-of-line images. A really large image may, indeed, be justified — as in a map — where detail matters. If you are only showing a photo of Monticello (Thomas Jefferson's home), that level of detail probably isn't necessary. You would be better off, in this case, to show the entire house in one photo and then show separate smaller images of the details you want to discuss.

### Out-of-line

When your visitors have to take additional action, such as click an image or agree to initiate a plug-in to see or hear your element, then it is an out-of-line element. Examples of out-of-line elements are images, sound, video, 3D models to be navigated, and animations.

Your visitors may get annoyed if they must take action too often to see or hear your multimedia. A good use of out-of-line sound is a music store that lets you click an icon to hear a music clip from an album. This enables the store to list many album titles on one page — obviously, there wouldn't be any way for you to hear the music clips from every album concurrently. Then the detail page from each album may have inline sound, with a clip from the most popular song on the album.

### Objects

Because the number of types of multimedia you can add to your page — many of them proprietary (meaning the technology is owned by only one company) — is growing faster than anyone can count, the WC3 isn't trying to keep up. Instead, it has defined a type of element, called an *object*, which you define when you want to refer to any of these proprietary multimedia types (RealAudio, Worlds, and Shockwave come immediately to mind).


Chapter 42 explains how to incorporate proprietary multimedia technologies into your page. It also introduces some of the more common "plug-ins" that Web sites use.

## Standardization

This section is being written in reverse order. First, you read about the rational process in place to ensure that every new technology is compatible with every existing technology and that all the major players in these industries sing songs holding hands. Then you read about the actual process an individual company goes through to bring a new technology to market before its competitors get it done. This includes all the compelling reasons why a company can't stop and wait for everyone else to agree (and catch up) with what it is doing. By the end

of this section, you understand why standardization rarely precedes technical innovation and why developers will always be in a position of having to make trade-offs between universal access to their page (by every browser that has ever been written) and implementing the latest, coolest technologies. Eventually, after a technology has aged a bit, all the players can come together and agree on a standard.

# Understanding the Standardization Process

The process of setting standards for Web-related technologies is run by the World Wide Web Consortium (W3C), based at MIT in the United States and at INRIA in Europe. The W3C is led by Tim Berners-Lee, "creator of the World Wide Web," as the W3C pages tell you repeatedly. The W3C is a consortium. The vocabulary used to define a *consortium*, how it runs, and how it works is a study in nonconfrontation. The person who runs it is a *director*. The people who get paid to do what Berners-Lee says are his *team*. The companies that pay to participate and get a vote in what is decided are *members*. When the W3C is thinking about setting a standard for a technology, it *initiates an activity*. When a W3C committee has decided upon what it thinks is a standard, it issues a *proposed recommendation*. After the membership votes to support the proposed recommendation, it becomes a *recommendation*.

No wonder the W3C doesn't claim to have any enforcement power. With a passive vocabulary like this, it couldn't enforce a bedtime!

Participation in the standardization practice is similar to participation in social activities. If you are going to show up, you'd better behave according to accepted norms.

## Activities

If a member of the W3C (members are universities, corporations, and organizations) thinks the W3C should look into setting a standard, then members can initiate an activity. Members may initiate activities both for technologies and for social issues affected by technology. If the W3C agrees some interest exists in this activity, then committees and mailing lists are formed to determine which direction that technology should go.

The Activity Proposal needs to answer some hard questions:

- What is the market? Who are the major players? Who stands to benefit from a standard? Are the players and the beneficiaries members of the W3C or will they join? Is the market new, growing, or a niche market? What alternatives to this technology are there? Who owns these technologies?
- How many resources of the W3C will this require?
- What is the scope of the project?
- Is there a deadline? What are the timelines?

- ✦ Are there any intellectual property conflicts?
- ✦ Is anyone else setting standards for this technology? Will there be a conflict? Whom will this affect?
- ✦ Is an activity already in place that covers this technology?

## Groups

If the director accepts an activity proposal, then a working group is formed to begin work on the activity. The goal of most *working groups* is to produce a statement or a proposed recommendation. The composition of the working group is important. Any technology company in any way affected by the recommendations regarding an activity will want to be part of this working group. You can bet both Netscape and Microsoft were represented on the HTML 4 working group. Only members can participate in working groups for activities.

## Consensus

Not surprisingly, the W3C operates by consensus. A working group must address the concerns of all participants and make an effort to resolve them before reaching a conclusion. A simple majority rule is not enough. The idea is the working group hashes through every possibility until either everyone is in agreement, the people who dissent are convinced the position arrived at is the best possible position given the constraints, or the people in the minority think the problem is no longer worth fighting over.

This philosophy protects small start-ups that come to the table with new technology to protect. It gives every party — whether Sun, Microsoft, or Midge's New Technology Company — equal weight.

## Proposed recommendations

Not all activity proposals result in proposed recommendations, but many do. When a working group has resolved the conflict raised by its membership and has drafted what it thinks is a pretty good attempt to set a standard, it publishes what is called a *proposed recommendation*. A proposed recommendation — along with working drafts that are developed along the way — is published to the W3C Web site. Most materials you purchase in bookstores explaining the newest technologies are actually written with the working drafts or the proposed recommendations as the reference materials.

## Voting

By following the process of consensus building that takes place during the drafting of the proposed recommendation, the hope is all major conflicts have been resolved by the time the proposed recommendation is issued. During the voting period, all

members of the W3C — whether or not they participated on the working group (and most wouldn't have) — vote on the proposed recommendation.

At the end of the voting period, the director, ever the facilitator of good will, has three choices:

1. Issue the proposed recommendation as a recommendation
2. Reject the proposed recommendation
3. Issue the proposed recommendation with comments

When members vote, they can make comments in addition to their vote. If the comments are substantial, especially if the comments that accompany no votes are substantial, then the director will not accept the proposed recommendation as a recommendation. Thus far, the process of consensus has been effective in dealing with the comments of members *before* the activity is no longer a draft; most proposed recommendations do become recommendations.

### Recommendations

Even after all these steps are taken to ensure that every member's viewpoint is heard, considered, and incorporated as much as possible, the conclusions regarding standardization that the W3C reaches are still only recommendations. Why? First, the W3C is an international standards body and does not yet have access to United Nations troops to enforce its resolutions. Second, all companies in the community of corporations are trying to develop the best new technologies without antagonizing *you*, the developer. Making up new and different ways to do the same thing, just to be different, doesn't serve the company's own interests. Each company is best served by having the largest pool of developers developing in that technology. This logic enticed Microsoft to offer native support for JavaScript in its Internet Explorer browsers. (Of course, Microsoft's native support isn't the same as Netscape's.) Companies that insist on "my way or the highway" often find themselves on the shoulder — alone.

## Players in the Standardization Process

So who gets a vote when the W3C asks? Members of the W3C. Membership in the W3C isn't exclusive, however. Any organization or company can join. Of course, a rather high membership fee exists, which pays the administrative costs of running the W3C.

Who would want to get involved in the W3C? Technology vendors, content providers, corporate users, research laboratories, standards bodies, universities, and governments. If you develop a new technology, you need to be at the table when the standards for related technologies are being set. Otherwise, you run the risk of being written right out of the standard. If your operations are affected by new

technologies, then you want to be on the working group, pushing for that standard in which you've already invested. If you develop content that makes use of new technologies, then you want to be well-positioned to know what resources you need to have in place when a new standard is implemented.

The major players, of course, are Microsoft and Netscape. They develop the browsers most widely used to visit Web sites. Microsoft and Netscape both want to see standards set that

- ✦ are backward-compatible with their own browsers
- ✦ make use of the newest technologies they are implementing — the way they are implementing them
- ✦ don't require them to add any features or elements coined by their competitors

But realistically, both browsers want to meet all the needs of the developers. If you, as a developer, find your site looks much better with one browser than with another, you may finally put a "Best Viewed with . . ." logo on your page and call it a day, rather than trying to make a silk purse out of a sow's ear with the other browser.

It might surprise you to learn Microsoft has been far quicker to adopt the technologies developed by others than has Netscape. Despite the bad rap Microsoft receives for its anticompetitive practices (so says Senator Arlen Specter), Microsoft implemented JavaScript as soon as it was clear this was what developers preferred (over its own VBScript). Netscape invented JavaScript, but it still hasn't opened everything in its browsers to scripting with the Document Object Model (DOM), the way Microsoft has.

# The Extensions Game

How do you get the standards set in your favor? You create extensions you know aren't standard and implement them on your browser. If they catch up in the developer community, the other browser maker will be forced to become compliant with the de facto standard you have set. At this point, if the W3C accepts it, this is just icing.

Consider the `FONT` element. The `FONT` element has never been part of an official W3C specification, but Netscape introduced it, Microsoft adopted it, and until CSS is fully and uniformly implemented in both browsers, the `FONT` element will continue to give the best results for formatting text available. At some point, standardization is irrelevant. Finally, with the HTML 4 specification, the W3C recognized the `FONT` element just enough to deprecate it!

# From Here


Proceed to Chapter 6 to learn about XML and XSL.

Go to Part III and begin learning about HTML structure in depth.

# Summary

HTML has come a long way from the days all pages had black, Times Roman text on a gray background. Today you can place text over graphics, layer images, and define paragraphs with indentation. With patience and an understanding of the standards currently in use, you can make your design visible as you intended to a majority of Web users.

Standards are a Web developers dream. Oh, for the day when you can implement a fabulous new technology and *know* it will work on all platforms on all browsers in the same way. But, alas, that day will never come. Browser makers and other purveyors of technology don't make money by finishing neck-and-neck with their competitors. They need to be the first one across the finish line to deliver on a new technology.

Even with the W3C spelling out standards as fast as they can, it will never be able to anticipate technology to such a degree that the standards precede the technology. All you can hope for is that within a reasonable period of time *after* a new technology is introduced, the competitive browser makers will implement it in a reasonably similar fashion. W3C or no W3C, the market is still the primary factor that drives standardization in this industry.

✦ ✦ ✦