CHAPTER 1

HTML 4 Basics

In This Chapter

An introduction to elements

Attributes: Having it your way

Entities: Using special characters

Adding comments

Standards for writing HTML

Avoiding syntax errors

Understanding nesting

Basic HTML structure

When you learn a language, you need to learn rules, called *syntax*. Unless you follow these rules, your statements don't have any meaning. Consider this statement, for instance: "Extra! Pizza, with a bring pepperoni me; cheese." The word order is out of whack and so is the punctuation. You can say this to a waiter, but good luck getting your pizza! Like any language, HTML has syntax rules. Fortunately, they're simple and easy to learn. Before you start trying to write HTML, you'll be wise to spend a little time learning these rules. It doesn't take long and there's a huge payoff: You'll avoid confusion and find it much easier to track down your errors.

This chapter introduces the basic building blocks of HTML, including elements, attributes, and entities. (Don't worry about defining these terms right now; that's done in this chapter.) You also learn the basics of nesting tags and about the basic, underlying structure of every HTML document. Once you learn these concepts, you're ready to starting writing HTML.

Introducing the Basic Building Blocks: Elements

When you write your HTML, you use *elements* to define the structure of the document, to define the presentation of your document, to define links to other documents, and to specify desired behavior. Examples of elements are: HEAD, BODY, P, BLOCKQUOTE, and UL. When you actually go to insert these elements into your text, you surround them with < (less than) and > (greater than) symbols, which are collectively referred to as *angle brackets*. Once you have done this, you have <HEAD>, <BODY>, <P>, <BLOCKQUOTE>, and <UL>. These are no longer called elements; they are now called *tags*.

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 over 400 member companies who want to have a say in the standards. The W3C tries to balance the interests of the academy, the companies 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 HTML, CSS, XML, and other essential technologies. If the W3C weren't maintaining a standard, all browsers might eventually be unable to talk to all Web servers. You can visit their Web site at `http://www.w3c.org`.

Elements versus tags

The World Wide Web Consortium (W3C) uses the word *elements* in two ways, which is rather confusing. You may have noticed this book does the same thing. In this chapter, we discuss elements in the *tag* sense of the word. The other kind of element is the *element of structure* of a document (for example, title, paragraph, blockquote).

Even elements have parts

Unlike the elements in the Periodic Table, elements in HTML usually have three parts: start tags, content, and end tags. Most elements have start tags and end tags. The *start tag* is the element name surrounded by angle brackets: `<HEAD>`, `<BODY>`, `<P>`, `<BLOCKQUOTE>`, and `<UL>`. The *end tag* is the element name, preceded by a / (called a *forward slash*), surrounded by angle brackets: `</HEAD>`, `</BODY>`, `</P>`, `</BLOCKQUOTE>`, and `</UL>`.

When the browser sees a start tag, it knows the text to come will all be of the type defined by the start tag. Not until the browser sees an end tag does it stop expecting the text to be of that type. Because elements can often be nested — not all elements can be nested within all other elements, but there are valid element nestings — it doesn't automatically assume a different start tag indicates the previous element type has ended. In fact, the browser assumes nothing. Forget about the benefit of the doubt. The browser takes everything you send it absolutely literally.

To add another layer of confusion, some elements will display correctly without their end tag, as long as the end of an element can be clearly determined from other surrounding elements. For example, the LI (list item) and P (paragraph) tag don't require end tags, since their end can be clearly determined by the beginning of the next element. Still, you won't go wrong by always including end tags.

Nesting. Placing elements within other elements. For example, in a table, the rows are nested within the table element, and the cells are nested within the row elements.

Recap: This stuff is really important!

Every element has a *name*.

The *start tag* is the element name surrounded by angle brackets.

The *end tag* always starts with a slash, has the element name, and is surrounded by angle brackets.

Most elements have content, which occurs between the start and the end tags.

Some elements have no content.

Some elements have no end tags.

Some elements don't take any content, such as `BR`, which forces a line break. Some elements have optional end tags, such as `LI` and `P`, which are list item and paragraph elements. When we explain an element, we tell you whether it has any content and whether it has a required end tag. Elements are case-insensitive: `<TITLE>`, `<Title>`, and `<title>` are all the same to the browser. To make reading your markup easier, though, we recommend you write your element names in all caps. This is the convention this book uses.

Between the start tag and the end tag, you place the *content*. In reality, you usually write the content first and put the start tag before your content, and then put your end tag after your content. A major cause of errors in HTML documents is forgetting the slash in the end tag. Be careful when you type your HTML and be sure you spell your tag names correctly, include both angle brackets, and include your slash. Three examples of syntactically correct HTML follow:

```
<H1>This is the <B>bolded title</B> of the page</H1>
<B>This statement will be in bold.</B>
<I>This statement will be in italics.</I>
```

Understanding Your Options: Attributes

Elements have attributes that give you flexibility in writing your HTML. Each element has its own unique attributes. You see patterns, but you can't just apply any old attribute to any old element.

Couples only

Attributes have values. In fact, they come in pairs. If you are going to include an attribute, you have to include the value for that attribute. The value for the attribute

is always enclosed in double quotation marks. Examples of attribute-value pairs are as follows:

```
align="center"
width="33%"
size="12"
name="first_name"
```

Always shop from a list

The values for some attributes come from a list of acceptable values that the W3C creates, when it sets the standards. In the case of *valign* (an attribute frequently used to tell the browser where on the page you want an image or a table to appear, relative to text), your choices are top, middle, bottom, and baseline. In the next chapter, you learn the shorthand for the element rules.

Please take a number

Some attributes take numbers as their values. In some cases, the numbers can be either a set number (usually of pixels, which are just the dots on your screen — comparable to dpi, the measurement for dots on your printer) or a percentage. For example, when you define a table, you may want the first column to be 25 percent of the screen width, the second column to be 50 percent of the screen width, and the third column to be 25 percent of the screen width. Regardless of how the screen is sized, it changes size to fit in proportionately. If you were to assume the screen was 636 pixels and divide it yourself, the last column of the table may not be visible if the visitor to your page didn't have the browser open in full-screen mode. For a form, however, when you indicate the *size* (one of the attributes for text fields) of a field, you are indicating the number of characters. For an image, when you indicate the *width* (one of the attributes for object elements), you are indicating the number of pixels.

The colors of the world

In both the HTML document itself and in the style sheet, some attributes take a color for a value. As you may have noticed, colors don't appear the same from one monitor to the next. You can, however, say a lot with colors. For example, take a page where everything is in seafoam green or watermelon; this page says *beach*.

Not only do colors not look the same from one monitor to the next, Macs and PCs actually use different system palettes. Chapter 37 covers this topic in detail.

Just as with everything else, an HTML way exists to convey information about colors. HTML gives you two choices: the name of the color (from their approved list) and the hex (hexadecimal) representation of the color. Only 16 named colors

exist, so if you are picky about the color you use, then you'll want to find the hex representation of that color.

Another thing you should know about colors on the computer is that colors are composed of red, green, and blue, thus the *RGB scale*. In hex, the first two digits are the amount of red, the next two are green, and the next two are blue. So FF0000 would be pure red. Did we mention the highest any digit can go is *F* ? We discuss this next. One more thing: 000000 is black; FFFFFF is white. That's not very intuitive, so you just have to remember when you add color to the screen, it gets lighter. Unlike paper, which is white when blank, the screen is black when blank.

When you indicate the value of an attribute is a color and you use hex representation, you need to precede the hex value with a # (pound sign). The 16 named colors are listed in Table 1-1 with their hex names. If you know you want a color between two of the colors in the chart, try selecting the value between the two hex values. First, you must know how to count in hex. Hex counts from 1 to 9, then A, B, C, D, E, and F. If you need to add 1 to F, you go to 10 (that's one-zero, not ten). Try this math problem along with us: 5 plus 6 is B. Plus 5 is 10. Plus 7 is 17. Minus 9 is E. That's not too hard, is it? Fortunately, we don't need to multiply or divide in hex!

Table 1-1
Color Codes in Hex

Color Name	***Hex Representation***
Black	#000000
Green	#008000
Silver	#C0C0C0
Lime	#00FF00
Gray	#808080
Olive	#808000
White	#FFFFFF
Yellow	#FFFF00
Maroon	#800000
Navy	#000080
Red	#FF0000
Blue	#0000FF
Purple	#800080
Teal	#008080
Fuchsia	#FF00FF
Aqua	#00FFFF

Recap: More important stuff!

Attributes are specific to the elements they modify.

Attributes always have values.

Attribute values must be enclosed in double quotation marks.

Attribute values can be one from a list, a number, a percentage, or a name of your creation.

Creativity counts

For some elements, the attribute value is something you make up yourself. For example, if you want to create a form and you want people to enter their first names, you assign the value of the attribute *name* to be "first_name." Normally, when you are working with fields, some server-side scripting is going on. Your field names should match the names you expect in your script.

Using Special Characters: Entities

Another example of the W3C's penchant for academic vocabulary, *entities* are simply characters you may want to display that don't appear on your keyboard or are characters with special significance to HTML (notably <,>,&, and "). The most common of these are (©) (copyright) and (™) (trademark). Three ways exist to write a code for an entity in your HTML. Whichever way you choose, you will find all entities begin with an ampersand (&) and end with a semicolon (;):

- Using character notation. While this is the easiest way to show an entity, there isn't a character notation for every single entity. For the most common ones, there are character representations. For example, `&copy;` is the entity for the copyright symbol. Character notation to indicate entities is one of the rare parts of HTML that is case-sensitive.
- Using decimal notation. The decimal representation of a copyright symbol is `&#169;`. Sure, you knew that!
- Using hex notation. Programmers dig this method. Instead of © to indicate the copyright symbol, they get to type `&#xA9;`, which is obviously way cooler. For the rest of the population, knowing A9 in hex (hexadecimal notation) equals 169 in the decimal system is useful. All hex numbers are preceded by a lowercase *x*.

Entities exist for many foreign languages, mathematical symbols, and English symbols that either can't be produced on a normal keyboard or are special to HTML.

For example, if you wanted to write "Good idea" in Greek, but the bulk of your document was English, you'd write:

```
&Kappa;&alpha;&lambda;&eta;&eta;&delta;&epsilon;&alpha;
```

which would render as: Καληιδεα.

Table 1-2 has the entities you will need most often, if you aren't planning to include foreign languages in your pages to any great extent. A complete list of entities can be found in Appendix D.

Table 1-2
Common Entities

Character notation	*Hex notation*	*Entity created*
		nonbreaking space
¡	¡	inverted exclamation mark
¢	¢	cent sign
£	£	pound sign
¤	¤	currency sign
¥	¥	yen sign
¦	¦	broken bar
§	§	section sign
¨	¨	Diaeresis
©	©	copyright sign
ª	ª	feminine ordinal indicator
«	«	left-pointing double-angle quotation mark
¬	¬	not sign
		discretionary hyphen
®	®	registered trademark sign
¯	¯	Macron
°	°	degree sign
±	±	plus or minus sign
²	²	superscript two

Continued

Table 1-2 *(continued)*

Character notation	*Hex notation*	*Entity created*
³	³	superscript three
´	´	acute accent
µ	µ	micro sign
¶	¶	paragraph sign
¹	¹	superscript one
º	º	masculine ordinal indicator
»	»	right-pointing double-angle quotation mark
¼	¼	fraction one quarter
½	½	fraction one half
¾	¾	fraction three quarters
¿	¿	inverted question mark

Adding Comments to Your HTML

You have spent so much time with your HTML, no way could you ever forget what you were thinking when you wrote it, right? Maybe, maybe not. Most people can't remember where they put the bank statement they meant to balance from last month. How can you expect to remember what you were thinking if you have to modify your HTML in six months?

Realistically, you want to use comments in your HTML to tell yourself — and anyone who inherits your files — basic information about what you are doing in there. For example, putting a comment line near the top of your document telling who created the file and on what date is pretty standard. If you expect your HTML to be a teaching tool at all, with people viewing the source to see how you did things, then you want to have especially helpful comments.

Begin your comment with <!--. Any text you put after the two dashes is comment. The browser will not even try to read it. Even if you use special characters, such as ampersands and slashes and quotation marks and angle brackets, your browser ignores everything it comes across until it sees -->. Your comments can span multiple lines. The browser doesn't care. It is only looking for -->.

Later, in Chapter 8, we discuss the use of special programs that increase your productivity when you write your HTML.

One of the nice features of most of these programs is colored tags. If your editor changes the color of the tags, then you can see visually whether you have remembered to close your comments — or whether the rest of your document will be processed as a comment (usually not what you want).

Making Your HTML Readable

Most of the markup of your HTML document (that is, the actual HTML elements that mark up your content) is case-insensitive. This means you can pretty much type your HTML as you please. You don't want to work this way, though, because you want your HTML to be readable both to you and to others. Consider the following HTML:

```
<html><head><title>Over the Web, Inc.</title></head><body>
<object type="img/gif" src="logo-f.gif" height="155" width
="90"><h1>List, Email, and Relationship Management 1.15</h1>
<p>Would you like to have a more effective and effortless way
to solicit your potential customer base on the Web?</p> <p><i>
Over the Web Stay in Touch&reg;</i> List, Email, and Relation
ship Management allows you to add a page to your Web site where
your customers can sign in. You can then send mail to them
based on the interests they indicate. The system is entirely
configurable by you. You can change the background color, add
your logo to the top of your page, and customize the interests
your customers indicate.</p>
```

What a mess! Even if you know what all the tags mean, you won't want to dive into that! Now consider:

```
<HTML>
<HEAD>
<TITLE>Over the Web, Inc.</TITLE>
</HEAD>
<BODY>
<OBJECT type="img/gif" src="logo-f.gif" height="155"
width="90">
<H1>List, Email, and Relationship Management 1.15</H1>
<P>Would you like to have a more effective and effortless way
to solicit your potential customer base on the Web?</P>
<P><I>Over the Web Stay in Touch&reg;</I> List, Email, and
Relationship Management allows you to add a page to your Web
site where your customers can sign in. You can then send mail
to them based on the interests they indicate. The system is
entirely configurable by you. You can change the background
color, add your logo to the top of your page, and customize
the interests your customers indicate.</P>
```

This is much friendlier on the eyes. We encourage you to make use of white space in your file (as well as in your pages, which we discuss later). The file won't be any larger (each line of blank space is only sent as one or two bytes of data).

Another convention you may have noticed is the use of all caps for element names and lowercase for attribute-value pairs. You needn't be fanatical about this. If the editor you use does something else, don't worry. Being consistent and leaving lots of white space when you write your markup, though, makes maintenance easier.

Avoiding Common Syntax Errors

Syntax isn't a tax on cigarettes and beer. *Syntax* is the rules of a language. In English, you don't even think about syntax. You know most sentences have a syntax of subject-verb-object. You know not to say (poets excluded): "Wrote the book me."

HTML has syntax, too. When you break the rules, it is politely referred to as a syntax error. When you sit down to start writing HTML, you may find one of the following in your page:

- ✦ Everything from the first italicized word on is italicized.
- ✦ You can see one of your tags in your browser.
- ✦ A form field does not appear, even though you know you created it in HTML.

Figure 1-1 shows some of these syntax errors appearing on a Web page.

Checklist for avoiding common errors when writing HTML

- ✓ Do you have angle brackets surrounding the start and end tags?
- ✓ Do you have a slash at the beginning of your end tag?
- ✓ Do you have quotation marks around the values of your attributes?
- ✓ Are all the attributes for your tags valid?

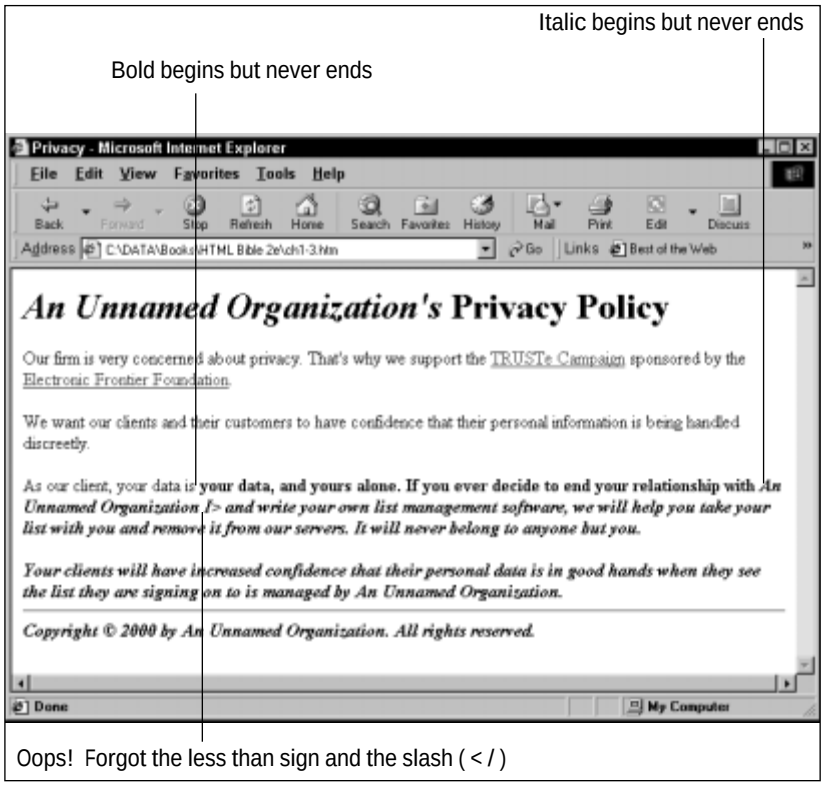

Figure 1-1: Common syntax errors

Understanding Nesting

As we discussed, sometimes you want to nest elements within other elements. For example, you may want to italicize something that occurs in a paragraph. You've seen the paragraph element, P, in all our examples. The italics element is I. In the previous code example (see "Making Your HTML Readable"), the words *Over the Web Stay in Touch* are italicized within a paragraph.

Many valid types of nesting exist. For example, nearly any element you put within the BODY element is valid, except for the HEAD element. You can't nest some elements, such as a link inside a link.

What is important to remember about nesting is you must close the inner element before you close the outer element. Consider this example of *invalid* HTML:

```
<H1>Over the Web Presents <I>Stay In Touch</H1></I>
```

The *valid* equivalent of this would be:

```
<H1>Over the Web Presents <I>Stay In Touch</I></H1>
```

Because the `I` element was the last one opened, it must be the first one closed. Be careful to close the inner element before you close the outer element. Most browsers know what to do with the previous invalid HTML, but as more and more elements become available, browsers will enforce rules more strictly. A day may come when the previous invalid HTML does not render properly.

The Basic Structure of an HTML Document: HEAD and BODY

A valid HTML 4 document has three parts:

1. **Version information.** This is also called the document type declaration used by this document. Earlier, we discussed that HTML is a document type declaration (DTD) of SGML. Three HTML DTDs exist; the version information should include which DTD you are using. If you are using an HTML editor, it takes care of this for you. If not, you will be safe using `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4-01 Frameset//EN""http://www.w3.org/TR/html4/frameset.dtd">`, which is the most permissive of the three.
2. **The HEAD.** The HEAD, in addition to being part of the HTML element, is an element of its own. The `HEAD` element can contain the title and meta data.
3. **The BODY.** Everything else you want to put into an HTML document belongs in the `BODY`. The BODY, like the `HEAD`, is also an element.

What we haven't mentioned yet is that HTML itself is an element. The HTML element has both a start tag and an end tag. The start tag should be placed after the version information and before the start tag of the HEAD element. The end tag goes at the very end of your document, after the end tag of the BODY element.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4-01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<HTML>
<HEAD>
<TITLE>My Valid HTML document</TITLE>
</HEAD>
```

```
<BODY>
If this had been an actual HTML document, this is where we
would have put the content.
</BODY>
</HTML>
```

The previous code example shows how a valid HTML document is structured. Notice how the BODY element is closed before the HTML element is closed.

From Here

For the more practical-minded, proceed to Chapter 2 to begin creating a Web page.

Summary

HTML syntax is straightforward. If you follow a few simple rules, you won't find your HTML document full of syntax errors and, more important, your pages will look right in a browser. The basic building block of HTML is an element. Elements have attributes that enable you to customize them. Attributes require values. Values should be enclosed in double quotation marks.

Most elements have both start tags and end tags. These tags surround the content. Both the start tags and end tags consist of the element name surrounded by angle brackets. The end tag has a slash before the element name, within the angle brackets.

An HTML document should have version information at the top. After that comes the HTML element, which consists of a HEAD and a BODY. To ensure you have nested elements properly, always close the inner element before you close the outer element.

✦ ✦ ✦