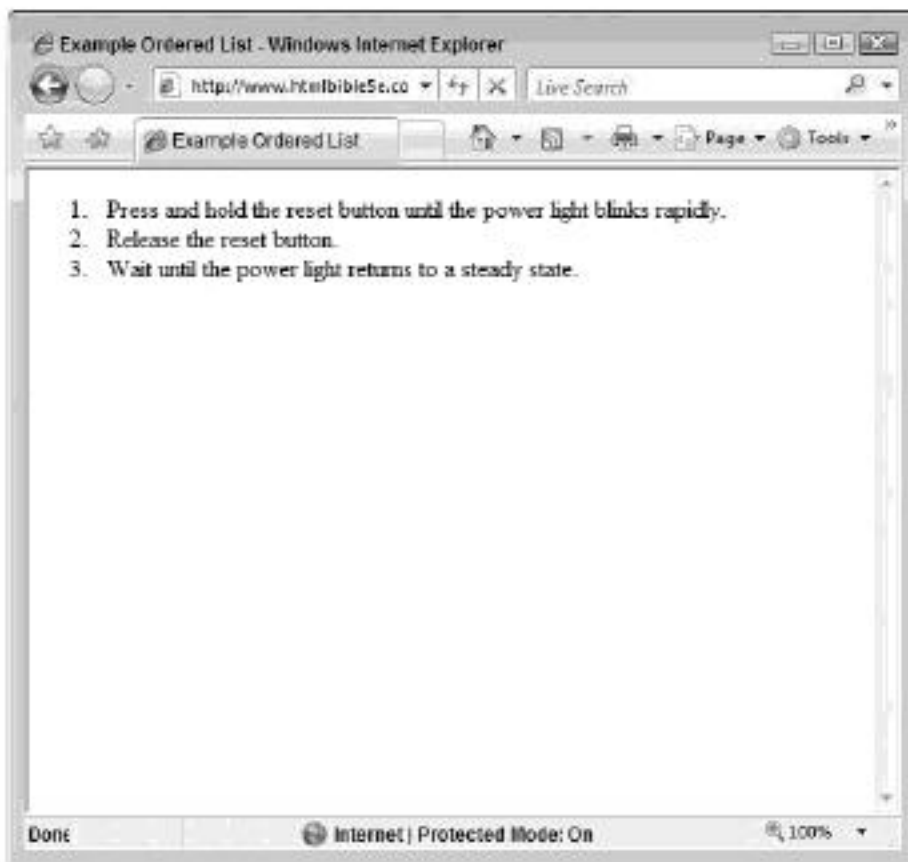


```
<head>
  <title>Example Ordered List</title>
</head>
<body>
  <p>
    <ol>
      <li> Press and hold the reset button until the power light blinks
        rapidly.</li>
      <li> Release the reset button.</li>
      <li> Wait until the power light returns to a steady state.</li>
    </ol>
  </p>
</body>
</html>
```

**FIGURE 7-1**

The default ordered list uses Arabic numbers for its items.



To specify a different type of identifier for each item, you use the `list-style-type` attribute and define a style for the list, as shown in the following code:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
```

```
<head>
  <title>Example Ordered List - Letters</title>
</head>
<body>
<p>
<ol style="list-style-type: upper-alpha">
  <li> Press and hold the reset button until the power light blinks
    rapidly.</li>
  <li> Release the reset button.</li>
  <li> Wait until the power light returns to a steady state.</li>
</ol>
</p>
</body>
</html>
```

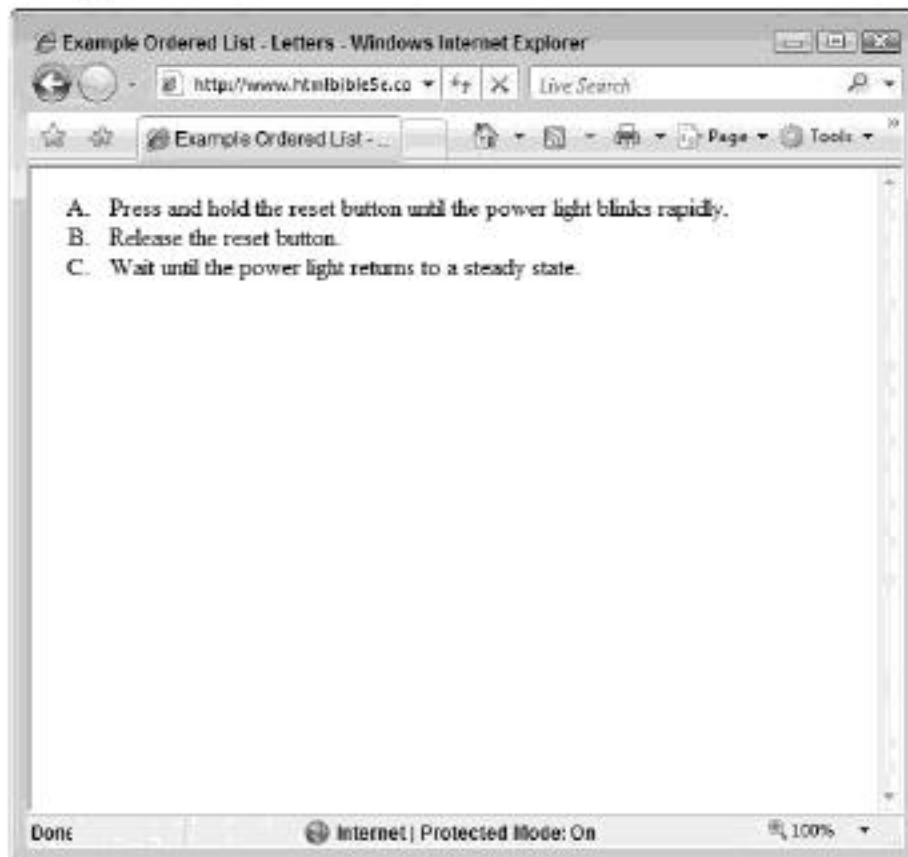
### Cross-Ref

Style properties for lists are covered in Chapter 31. ■

This code results in the list items being prefaced with uppercase letters, as shown in Figure 7-2.

**FIGURE 7-2**

The upper-alpha value of the list-style-type attribute causes the ordered list elements to be prefaced with uppercase letters.



### Note

Using letters or Roman numerals only makes sense for organizational lists (outlines, and so on), not for lists that outline a series of steps that must be followed in order. ■

The `list-style-type` property supports the following values in CSS2:

- decimal
- decimal-leading-zero
- lower-roman
- upper-roman
- lower-greek
- lower-alpha
- lower-latin
- upper-alpha
- upper-latin
- Hebrew
- Armenian
- Georgian
- cjk-ideographic
- hiragana
- katakana
- hiragana-iroha
- katakana-iroha
- none

### Note

Some of the values for `list-style-type` are font-dependent, meaning they are supported on certain fonts only. If you are using a type such as `hiragana` with a Latin-based font, you will not achieve the results you intended. ■

The values for `list-style-type` are self-explanatory. The default type is typically decimal, but it can be defined by the individual user agent. Keep in mind that your document's font and language options must support the language character sets used by the `list-style-type`.

Ordered lists also support the `list-style-position` property. This property controls where the number or character preceding each item appears. The property has two possible values:

- `outside` — The number or character appears outside the left margin of the item text.
- `inside` — The number or character appears inside the left margin of the item text.

The default is `outside`, and the difference between the two options is shown in Figure 7-3.

### Tip

The various list properties can all be defined within one property, `list-style`. The `list-style` property has the following syntax:

```
list-style: <list-style-type> <list-style-image>  
          <list-style-position>
```

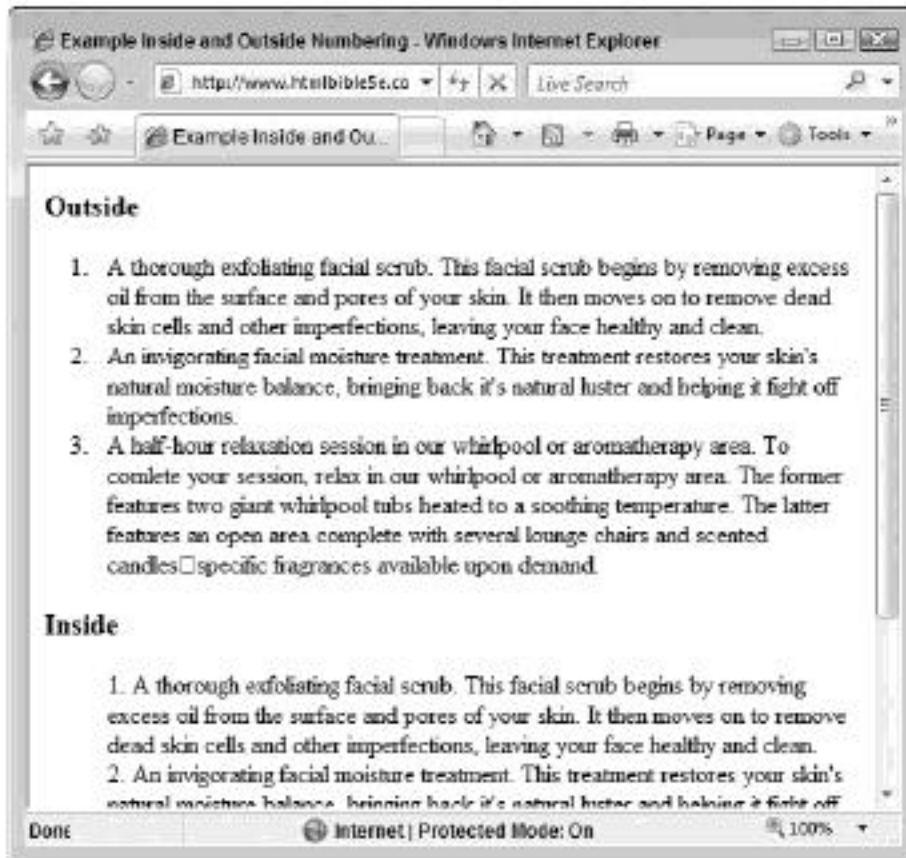
You can use this one property to specify one, two, or all three `list-style` properties in one declaration. For example, to define an ordered list with lowercase letters and inside positioning, you could use the following tag:

```
<ol style="list-style: lower-alpha inside;">
```

See Part III of this book for more information on styles. ■

**FIGURE 7-3**

The list-style-position property controls where the list item numbers/characters appear — outside or inside the list item margins.



## Changing the Start Value of Ordered Lists

Previous versions of HTML allowed the use of the `start` attribute in the `<ol>` tag to control what number or letter the list began with. For example, the following code starts a list with the decimal number 12:

```
<ol start="12" style="list-style-type: decimal;">
```

However, the `<ol>` tag's `start` attribute was deprecated and a replacement CSS equivalent has yet to be defined. Although you can use the `start` attribute, your document will no longer validate against *strict* HTML.

If you find yourself needing consistent yet flexible numbering, consider using the new CSS2 automatic counters and numbering feature. This feature uses the `content` property along with the new `counter-increment` and `counter-reset` properties to provide a fairly flexible automatic counter function.

The following style code defines a counter and causes any `<ol>` list to begin with an item number of 11:

```

<style type="text/css">
ol.eleven { counter-reset: list 10; }
li { list-style-type: none; }
li:before {
    content: counter(list,decimal) ". ";
    counter-increment: list; }
</style>

```

This code introduces quite a few CSS2 concepts: pseudo-elements, counters, and related properties and methods. However, it isn't as complex as it might first appear:

- The `ol` definition causes the counter (`list`) to be reset to 10 every time the `<ol>` tag is used — that is, at the beginning of every ordered list with a class of `eleven`.
- The `li` definition sets the list style type to `none` — the counter will display our number; if we left the `list-style-type` set to `decimal`, there would be an additional number with each item courtesy of the tag itself.
- The `li:before` definition does two things: It causes the counter to be displayed before the item (using the `before` pseudo-element and the `content` property) along with a period and a space. It increments the counter. Note that the counter increment happens first, before the display. That is why you need to reset the counter to one lower than your desired start.

Using the preceding styles along with the following list code in a document results in a list with items numbered 12–15:

```

<ol class="eleven">
  <li>Item 11</li>
  <li>Item 12</li>
  <li>Item 13</li>
  <li>Item 14</li>
</ol>

```

Counters are a relatively new feature of CSS2. Unfortunately, at the time of this writing, few user agents fully support counters. However, the other browsers are sure to follow suit at some point. Until then, you might consider using JavaScript or another client/server scripting method to generate dynamic numbers. You'll find more information on counters and the `content` property in Chapter 35.

## Unordered (Bulleted) Lists

Unordered lists are similar to numbered lists except that they use bullets instead of numbers or letters before each list item. Bulleted lists are generally used when providing a list of nonsequential items. For example, consider the following list:

- Action
- Role Playing
- Puzzle
- Adventure

## Part I: Creating Content with HTML

---

Unordered lists use the unordered list tag (<ul>) to delimit the entire list and the list item tag (<li>) to delimit each individual list item.

In the preceding example, the list has four elements, each preceded by a small, round, filled bullet. This is the default for unordered lists in HTML, as shown in the following code, whose output is shown in Figure 7-4:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
  <title>Example Unordered List</title>
</head>
<body>
<p>
<ul>
  <li>Action</li>
  <li>Role Playing</li>
  <li>Puzzle</li>
  <li>Adventure</li>
</ul>
</p>
</body>
</html>
```

Unordered lists also support the `list-style-type` property, but with slightly different values:

- `disc`
- `circle`
- `square`
- `none`

The default bullet type is `disc`, although the client browser can define the default differently. The different bullet types are shown in Figure 7-5.

As with ordered lists, you can define the `list-style-position` property, which in the case of unordered lists controls where the bullet appears — outside or inside the left margin of the item. For example, to move the bullet inside the item's margins, use a style with the <ul> tag, similar to the following:

```
<ul style="list-style-position: inside;">
```

Unordered lists support one other type of bullet for each item, an image. An image for use in unordered lists must fit the following criteria:

- Be accessible to the document via HTTP (be on the same Web server or accessible from another Web source)
- Be in a suitable format for the Web (JPEG, GIF, or PNG)
- Be sized appropriately for use as a bullet

**FIGURE 7-4**

Example of an unordered list



To specify an image for the list, use the `list-style-image` property. This property has the following syntax:

```
list-style-image: url(url_to_image);
```

This property can be used to add more dimension to standard bullets (for example, creating spheres to use instead of circles) or to use specialty bullets that match your content.

Of course, the graphics must be scaled to an appropriate “bullet” size and saved in a Web-friendly format. In the following example, two images were reduced to 10–20 pixels square and saved on the Web server as `sphere.jpg` and `sight.jpg`. The code uses the images as bullets, and the output is shown in Figure 7-6:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
  <title>Example Unordered List with Image Bullets</title>
</head>
<body>
<p><b>sphere</b></p>
```

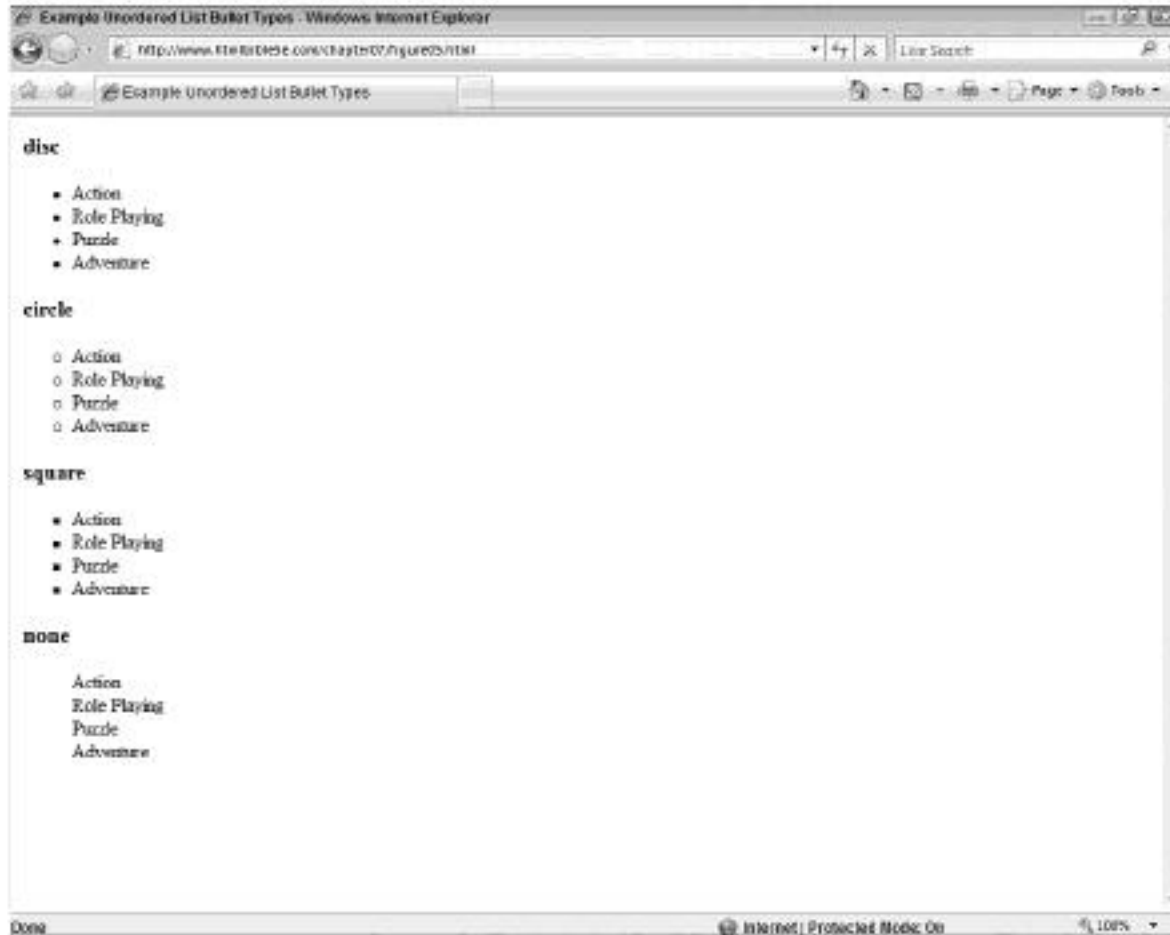
## Part I: Creating Content with HTML

---

```
<p>
<ul style="list-style-image: url(sphere.jpg);">
  <li>Action</li>
  <li>Role Playing</li>
  <li>Puzzle</li>
  <li>Adventure</li>
</ul>
</p>
<p><b>sight</b></p>
<p>
<ul style="list-style-image: url(sight.jpg);">
  <li>Action</li>
  <li>Role Playing</li>
  <li>Puzzle</li>
  <li>Adventure</li>
</ul>
</p>
</body>
</html>
```

**FIGURE 7-5**

Different bullet types for unordered lists



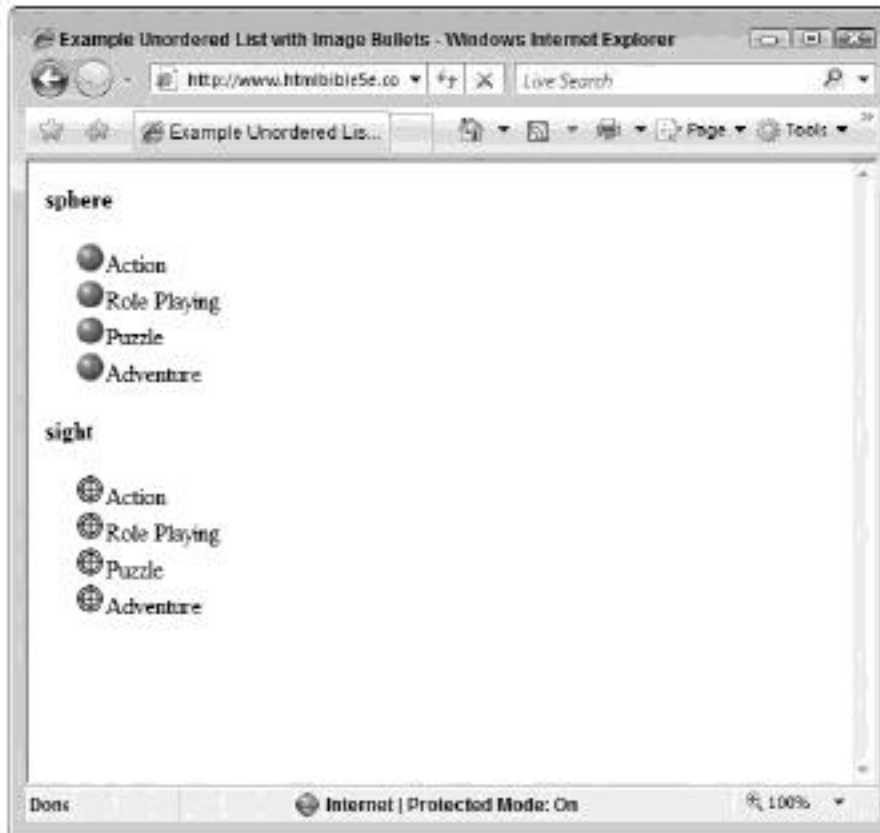


### Note

A few references state that the closing item tags (`</li>`) are not necessary in lists. Although most browsers will render the list properly without them, your code will not validate against strict HTML unless you include them. ■

**FIGURE 7-6**

Using graphic images for bullets in unordered lists



## Definition Lists

Definition lists are slightly more complex than the other two types of lists because they have two elements for each item: a term and a definition. However, there aren't many formatting options for definition lists, so their implementation tends to be simpler than that of the other two lists.

Consider this list of definitions:

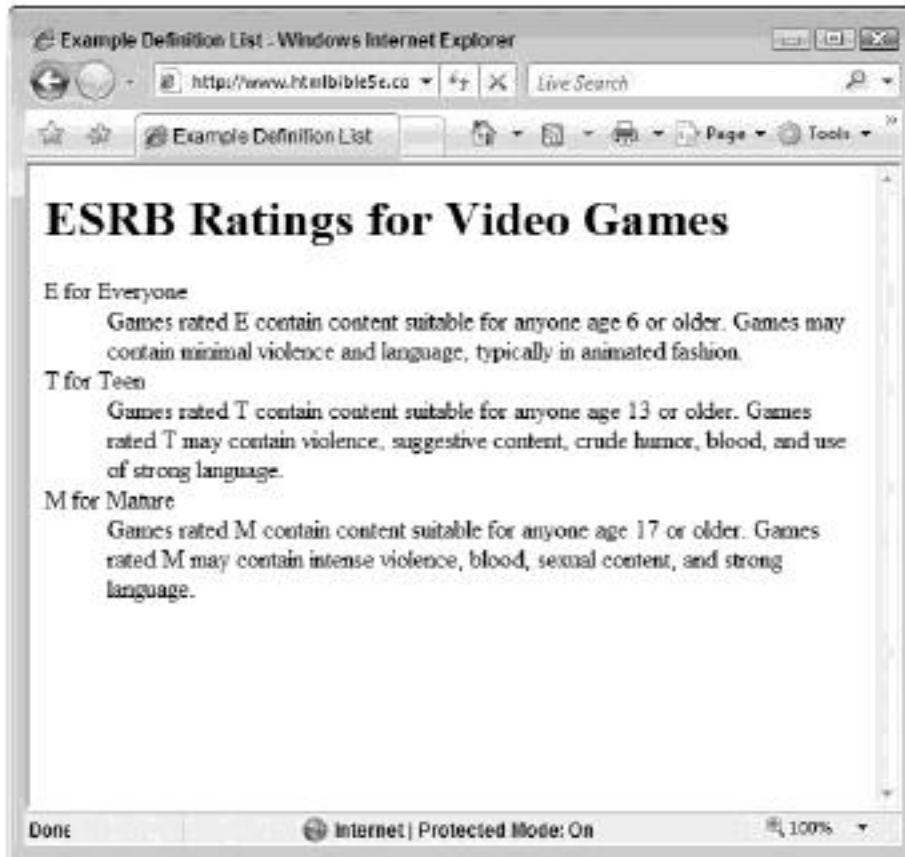
*E for Everyone.* Games rated E contain content suitable for anyone age 6 or older. Games may contain minimal violence and language, typically in animated fashion.

*T for Teen.* Games rated T contain content suitable for anyone age 13 or older. Games rated T may contain violence, suggestive content, crude humor, blood, and use of strong language.

*M for Mature.* Games rated M contain content suitable for anyone age 17 or older. Games rated M may contain intense violence, blood, sexual content, and strong language.

**FIGURE 7-7**

Definition lists provide term and definition pairs for each list item.



The definition items can be coded as list terms and their definitions as list definitions, as shown in the following code. The output of this code is shown in Figure 7-7.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
    <title>Example Definition List</title>
</head>
<body>
<h1>ESRB Ratings for Video Games</h1>
<p>
<dl>
    <dt>E for Everyone</dt>
    <dd>Games rated E contain content suitable for anyone age 6
        or older. Games may contain minimal violence and language,
        typically in animated fashion.</dd>
    <dt>T for Teen</dt>
    <dd>Games rated T contain content suitable for anyone age 13
        or older. Games rated T may contain violence, suggestive
```

```
        content, crude humor, blood, and use of strong
        language.</dd>
    <dt>M for Mature</dt>
    <dd>Games rated M contain content suitable for anyone age 17
        or older. Games rated M may contain intense violence,
        blood, sexual content, and strong language.</dd>
</dl>
</p>
</body>
</html>
```

## Note

To add clarity to your definition lists, construct styles that set the definition term in a different font or textual style. For example, you might want the definition terms to be red, bold, and italic. The following style definition accomplishes this:

```
<style type="text/css">
    dt { color: red; font-style: italic;
        font-weight: bold }
</style> ■
```

---

## Nested Lists

You can nest lists of the same or different types. For example, suppose you have a bulleted list and need a numbered list beneath one of the items, as shown:

- Send us a letter detailing the problem. Be sure to include the following:
  1. Your name
  2. Your order number
  3. Your contact information
  4. detailed description of the problem

In such a case, you would nest an ordered list inside an unordered one, as shown in the following code:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
    <title>Example Definition List</title>
</head>
<body>
<p>
<ul style="list-style-type: disc;">
    <li>Send us a letter detailing the problem. Be sure to
        include the following:</li>
    <ol style="list-style-type: decimal;"> <li>Your name.</li>
        <li>Your order number.</li>
```

## Part I: Creating Content with HTML

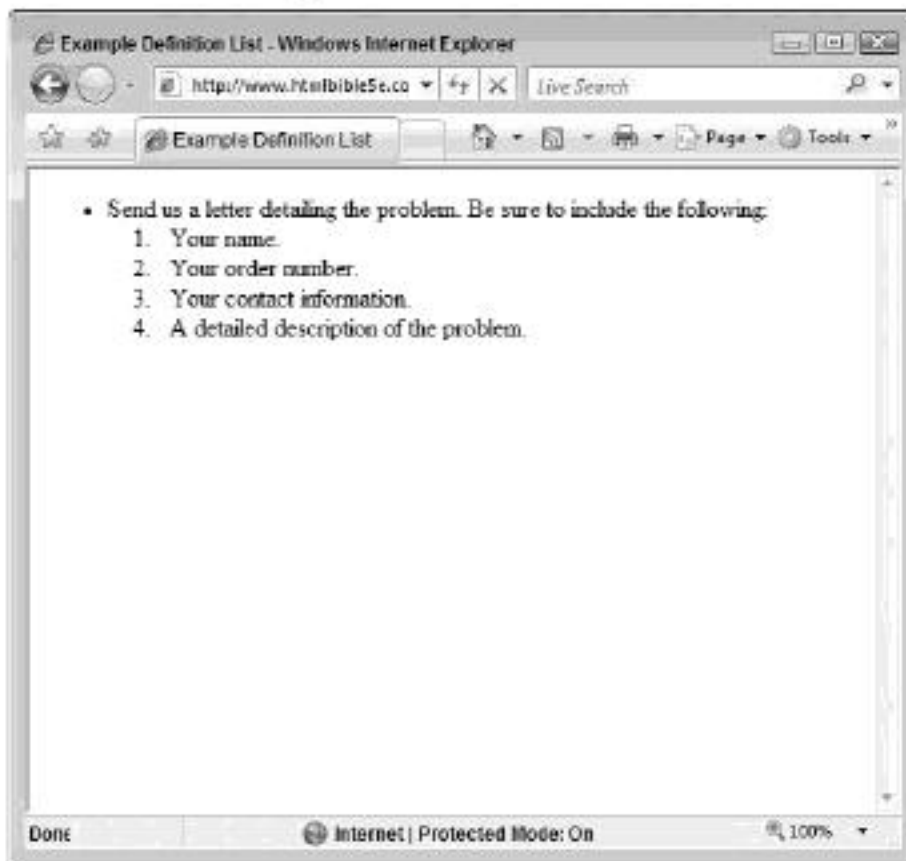
---

```
</li>Your contact information.</li>
<li>A detailed description of the problem.</li>
</ol>
</ul>
</p>
</body>
</html>
```

The code's output is shown in Figure 7-8.

**FIGURE 7-8**

You can nest different types of lists within one another.



Note that the nested list does not span any open or close tags — it starts after the close tag of the parent's item and before any other tags in the parent list. It is also formatted (indented) to make it easier to identify in the code. Using this method, you can nest any list within any other list.

### Summary

---

This chapter covered the ins and outs of the three different list types in HTML: numbered, bulleted, and definition. You learned how to define and format each type of list and how you can nest lists for more flexibility.

From here, you should work through the rest of the HTML formatting essential chapters (Chapters 8 through 12), covering links, tables, frames, forms, and more. This will help round out your knowledge of how to format document elements using HTML. Furthermore, reading the CSS basic chapters (Chapters 25 through 28) will give you the basics to apply to the specific CSS concepts in Chapters 29 through 38.



# Links

**L**inks are what make the World Wide Web weblike. One document on the Web can link to several other documents, and those in turn link to other documents, and so forth. The resulting structure, if diagrammed, resembles a web. The comparison has spawned many “web” terms commonly used on the Internet; for example, electronic robots that scour the Web are known as *spiders*.

Besides linking to other documents, you can link to just about any content that can be delivered over the Internet — media files, e-mail addresses, FTP sites, and so on.

This chapter covers the ins and outs of linking to references inside and outside the current document and how to provide more information about your documents’ relationships to others on the Web.

## What’s in a Link?

Web links have two basic components: the *link* and the *target*, or *destination*.

- The link is the tag in the main document (source) that refers to another document.
- The target, or destination, is the document (or particular location in the document) to which the link leads.

For example, suppose the Acme Games website reviews video games, and the site posts an extremely positive review of a game by On Target Games. Acme could put a link in the review on its site, leading to the game’s product page on On Target’s site. Such an arrangement would resemble the diagram shown in Figure 8-1.

### IN THIS CHAPTER

What’s in a Link?

Linking to a Web Page

Absolute versus Relative Links

Link Targets

Link Titles

Keyboard Shortcuts and Tab Order

Creating an Anchor

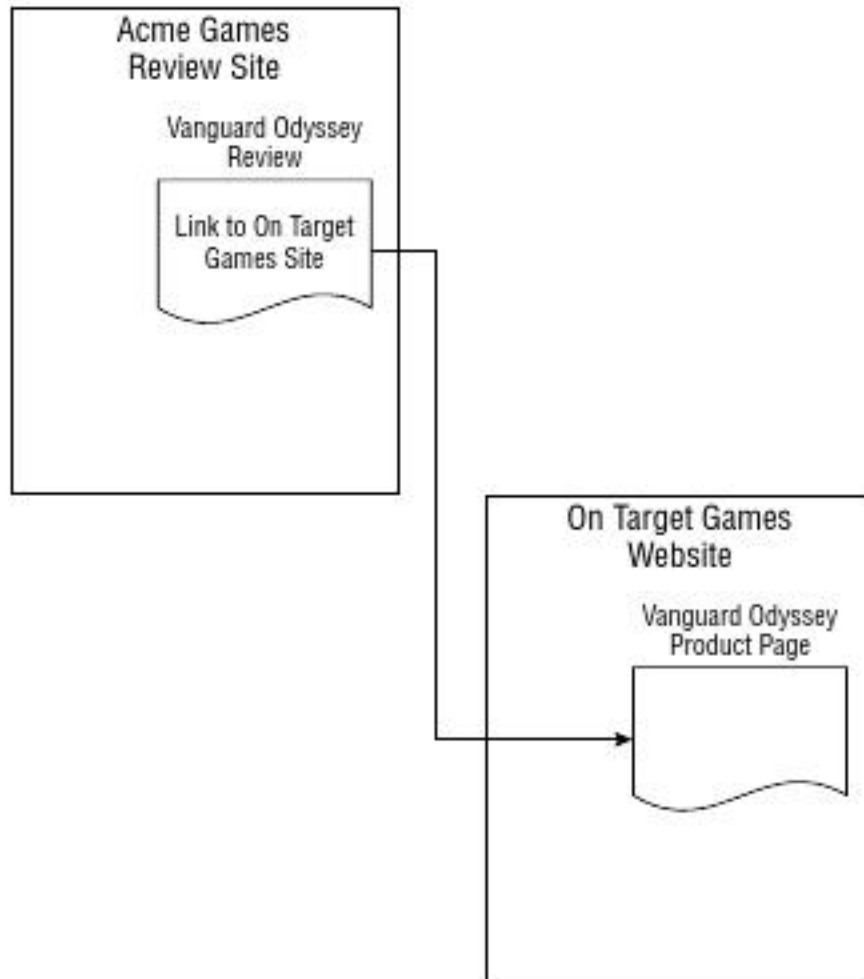
Choosing Link Colors

Link Destination Details

The Link Tag

**FIGURE 8-1**

The relationship of documents on the Web via links — the user clicks the link in the review document to reach the specified page on the On Target Games website.



Links have two components: a descriptor and a reference to the target. The target is a document that can be delivered via the Internet. In the preceding example, the review might list the manufacturer's name as the descriptor, and the actual Web URL would be the reference. Both are specified in the anchor tag (`<a>`), as follows:

```
<a href= "url_of_target">descriptor</a>
```

The target reference is specified via the `href` attribute, and the descriptor appears between the start and end anchor tags. For example, a link to On Target Games would resemble the following, if the domain for On Target is `on-target-games.com`:

```
<a href="http://www.on-target-games.com">On Target Games' Website</a>
```

If you don't provide the name of a document in the link, the Web server (in this case, `on-target-games.com`) will send the defined top-level document (known as an *index document*). Typically, this document is named `index.html` or `home.html`. If such a document



doesn't exist or one has not been defined for the server, the server will either display an index page or send a "not found" error to the client's user agent.

The text "On Target Games' Website" would be highlighted in the document to indicate it is a link. The default highlight for a link is a different color font and underlined — you will see how to change the highlight later in this chapter.

### Note

As mentioned in the introduction to this chapter, you can link to other things besides HTTP documents. All you need is the URL of the item to which you wish to link and the protocol necessary to reach the item. For example, if you wanted to link to a document on an FTP site, you could use an anchor tag similar to the following:

```
<a href="ftp://ftp.on-target-games.com/demos/vanguarddemo.zip">
Vanguard Demo</a> ■
```

Note that the protocol is specified (ftp: instead of http:) and the server name is specified (ftp.on-target-games.com), as is the path and filename (demos and vanguarddemo.zip). A similar method can be used to link to an e-mail address (href="mailto:someone@on-target-games.com"). Clicking such a link will generally spawn the user's e-mail client, ready to send an e-mail to the address specified.

### Note

The rest of this chapter concentrates on linking to HTML documents on the Web. However, all the concepts addressed here apply when linking to other content types. ■

## Linking to a Web Page

---

The most popular link style on the Web is a link to another Web page or document. When activated, such a link causes the target page to load in the user agent. Control is then transferred to the target page — its scripts run, and so on.

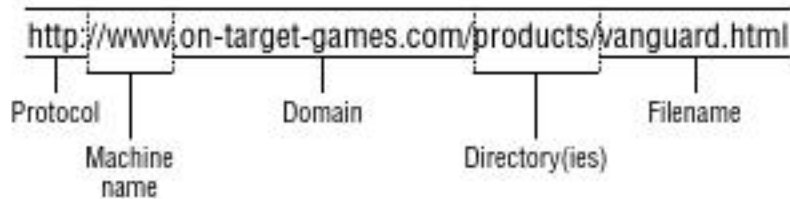
To link to another page on the Internet, simply specify the target's URL in the anchor tag. Suppose you want to link to the products page of the On Target Games Website and the page is named `vanguard.html` and resides in the `products` directory on the Web server. The `href` parameter of the link would be as follows:

```
http://www.on-target-games.com/products/vanguard.html
```

Note that the URL contains the protocol, the server name, the directory name, and the filename. Figure 8-2 shows a breakdown of the various pieces of the URL.

**FIGURE 8-2**

The various pieces of a URL



In the case of this URL, the various pieces are separated by various key characters:

- The protocol is first, and ends with a colon (`http:`).
- The server name is next, prefaced with a double slash (`//www.on-target-games.com`).
- The directory (or directories) comes next, separated with slashes (`/products/`).
- The page's filename is last, separated from the preceding directory by a slash (`/vanguard.html`).

### Note

The server name is actually two pieces: the server's name and the domain on which it resides. In `www.on-target-games.com`, `www` is the server name and `on-target-games.com` is the domain. There is a common misconception that all Web server names need to begin with `www`. Although `www` is a standard name for a Web server, the name can be almost anything. For example, the U.S.-based Web server for the Internet Movie Database (`imdb.com`) is `us.imdb.com`. ■

## Absolute versus Relative Links

---

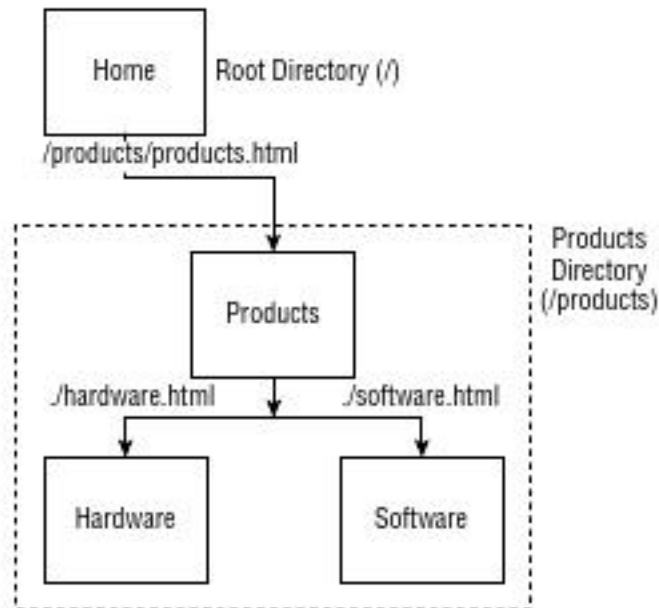
There are two types of URL styles, and therefore two link types that you need to understand: *absolute links* and *relative links*. You have seen absolute links, where the URL used in the link provides the full path, including the protocol and full server address. These links are called *absolute links* because the URL itself is absolute — that is, it does not change no matter where the document in which it appears is kept.

The other type of link, a *relative link*, does not provide all of the details to the referenced page; hence, its address is treated as relative to the document where the link appears. Relative links are useful only for linking to other pages on the same website because any reference off of the same site requires the remote server's name.

Suppose you are the Webmaster of a company website on the Internet. You have several pages on the site, including the home page, a main products page, and hardware and software products pages. The home page is in the root directory of the server, while the product pages (all three) are in a products directory. The relative links back and forth between the pages are shown in Figures 8-3 and 8-4.

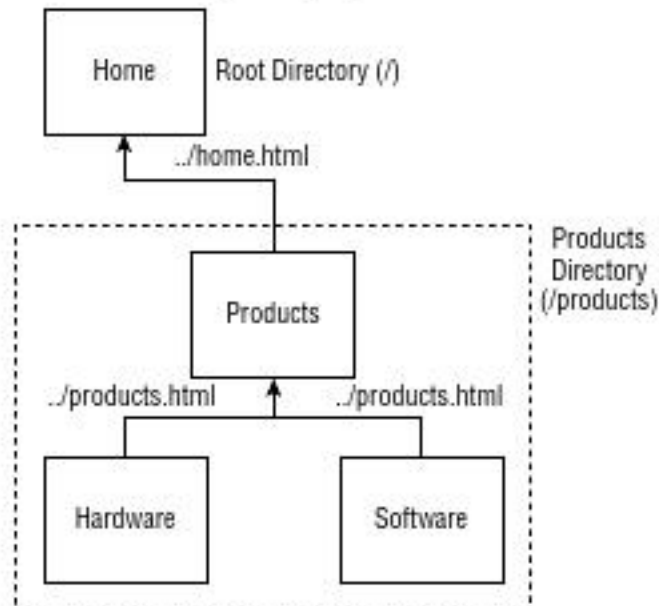
**FIGURE 8-3**

Relative links to subpages



**FIGURE 8-4**

Relative links to parent pages



Note that you can use directory shortcuts to specify where the pages are:

- Starting a directory with a slash (/) references it in a subdirectory of the root directory.
- Starting a directory with a period and a slash (./) references it in a subdirectory of the current directory (the directory where the current page resides).
- Starting a directory with a double period and a slash (../) references it in a parent directory to the current directory.

## Part I: Creating Content with HTML

Relative links are easier to maintain on sections of websites where the pages in that section never change in relationship to one another. For example, in the case of the site shown in Figures 8-3 and 8-4, if the products pages move as a whole unit to another place on the site, the relative links between the product pages won't change. If the links were coded as absolute (for example, `http://www.domain-name/products/hardware.html`), they would all have to change.

## Link Targets

Normally, links open the page they refer to in the active user agent window, replacing the page currently displayed. However, you can control where the page opens using the `target` attribute in the anchor tag.

### Note

The `target` attribute has been deprecated in strict HTML because it directs the destination of a link to be opened in a specific frame target. Because frames have been deprecated in the strict version of HTML 4.01, so has the `target` attribute. It appears here because most user agents still support the attribute and it can be useful. However, keep in mind that using this attribute means your documents will not validate against strict HTML. ■

The `target` attribute supports the values shown in Table 8-1.

**TABLE 8-1**

### Target Attribute Values

Value	Description
<code>_blank</code>	Opens the linked document in a new window.
<code>_self</code>	Opens the linked document in the same frame as the link.
<code>_parent</code>	Opens the linked document in the parent frameset.
<code>_top</code>	Opens the linked document in the main window, replacing any and all frames present.
<code>name</code>	Opens the linked document in the window with the specified name.

### Cross-Ref

Frames are covered in Chapter 10. ■

For example, to open a linked document in a new window, rather than replace the contents of the current window with the linked document, you would use a tag similar to the following:

```
<a href="http://www.oasisoftranquility.com" target="_blank">
Monthly Drawing (new window)</a>
```

## Caution

The debate about whether you should ever open a new window is fierce. Most users are accustomed to all new windows being of the pop-up ad variety — and very unwelcome. However, from a user interface standpoint, new windows can be utilized very effectively if they are used like dialog boxes or new windows that an operating system spawns. In any case, you should form a habit of informing users when you are going to open a new window so you don't surprise them. ■

The last value listed for target, *name*, can also aid in the user interface experience, if used correctly. Certain methods of opening windows (such as the JavaScript `window.open` method) enable you to give a window a unique name. You can then use that name to push a linked document into that window. For example, the following code displays two links; the first opens a new, empty user agent window named `NEWS`, and the second pushes the content at `www.yahoo.com` into the window:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<script type="text/javascript">
function NewsWindow(){
    var fin=window.open("", "NEWS", "width=400,height=400");
}
</script>
</head>
<body>
<p><a href="#" onclick="NewsWindow()">Open a Blank Window</a></p>
<p><a href="http://www.yahoo.com" target="NEWS">Fill Window with
Yahoo Content</a></p>
</body>
</html>
```

## Cross-Ref

For more information on JavaScript, refer to Chapter 16. ■

---

## Link Titles

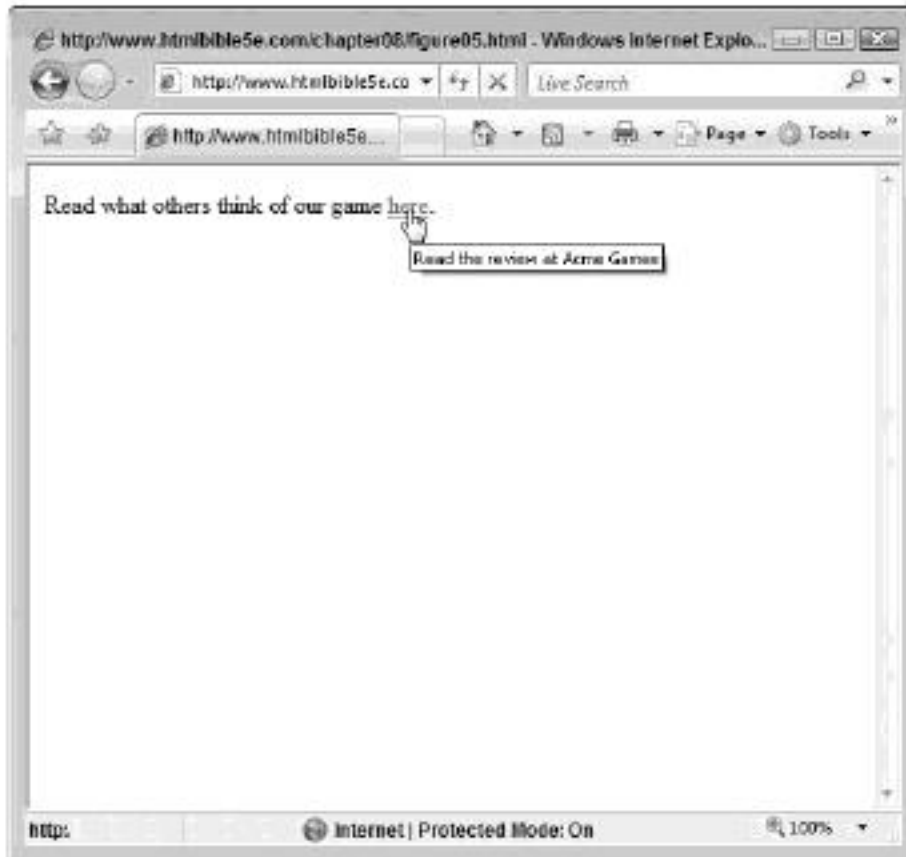
You can also title a link using the `title` attribute in the anchor tag. This causes most current user agents to display the title text as a tooltip when the mouse hovers over it. For example, the following link will cause Internet Explorer to display “Read the review at On Target Games,” as shown in Figure 8-5:

```
Read what others think of our game <a
href="http://www.on-target-games.com/reviews/vanguard.com"
title="Read the review at On Target Games">here</a>.
```

You can use this feature to give users more information about a link before they click it.

**FIGURE 8-5**

The title attribute causes a tooltip display when the mouse hovers over the link.



## Keyboard Shortcuts and Tab Order

---

In the modern world of computers it is easy to make assumptions about users, their hardware, and capabilities. Several years ago, no one would have dreamt of delivering rich, multimedia content over the Web. Today, however, it is often assumed that everyone is using the latest user agent, on a high-end computer, across a broadband connection.

That isn't always the case. In fact, some users who visit your site may not even have a mouse to aid in browsing. The reason could be a physical handicap, a text-only agent, or just a fondness for using the keyboard. You can accommodate these users by adding additional methods to access links on your page.

### Keyboard shortcuts

Each link can be assigned a shortcut key for easy keyboard-only access using the `accesskey` attribute with the anchor tag. The `accesskey` attribute takes one letter as its value: the letter that can be used to access the link. For example, the following link defines “R” as the access key:

```
<a href="http://www.on-target-games.com/reviews"
accesskey="R"><b>R</b>reviews</a>
```

Note that different user agents and different operating systems handle access keys differently. Some user agent and operating system combinations require special keys to be pressed with the defined access key. For example, Windows users on Internet Explorer must hold the Alt key down while they press the access key. Note also that different user agents handle the actual access of the link differently. Some user agents will activate the link as soon as the access key is pressed, while others only select the link, requiring another key to be pressed to actually activate it.

### Tip

**Keyboard shortcuts won't help your users if you don't give them a clue as to what the shortcut is. In the example earlier in this section, the defined shortcut key ("R") was used in the link text and highlighted using the bold font attribute. ■**

## Tab order

Defining a tab order for the links in your document will also help your users. As with most graphical operating systems, the Tab key can be used to move through interface elements, including links. Typically, the default tab order is the same as the order in which the links appear in the document. However, on occasion, you might wish to change the order using the `tabindex` attribute. The `tabindex` attribute takes an integer as its value, and that integer is used to define the tab sequence in the document. For example, the following code switches the tab order of the second and third links in a document:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Tab Ordered Document</title>
</head>
<body>
<p>This is the <a href="http://www.example.com"
tabindex="1">first link</a>.</p>
<p>This is the <a href="http://www.example.com"
tabindex="3">second link</a>.</p>
<p>This is the <a href="http://www.example.com"
tabindex="2">third link</a>.</p>
</body>
</html>
```

### Note

**As with most interface elements in HTML, the user agent defines how `tabindex` is implemented and how tabbed elements are accessed. ■**



# Creating an Anchor

---

Anchor tags have another use; they can be a marker in the current document to provide a bookmark that can be directly linked to. For example, a large document might have several sections. You can place links at the top of the document (or in a special navigation frame) to each section, enabling the user to easily access each section.

To create an anchor in a document, use the anchor tag with the `name` attribute. For example, the following code creates an introduction anchor at the “Introduction” heading:

```
<h1><a name="introduction">Introduction</a></h1>
```

To link to the anchor you use a standard link, but add the anchor name to the end of the URL in the link. To identify the name as an anchor, separate it from the rest of the URL with a pound sign (#). For example, suppose the `introduction` anchor appears in the document `vanguard.html`. To link to the `introduction` anchor, you could use the following code:

```
<a href="vanguard.html#introduction">Introduction</a>
```

## Note

Because the URL in the link tag can contain the server and document names as well as the anchor name, you can link to anchors in the same document or any accessible document. If you are linking to an anchor in the same document, you can use an abbreviated form of the URL that includes only the pound sign and the anchor name. ■

In addition to using the anchor tag for bookmarks, you can link to a block element’s `id` attribute. For example, if the `Introduction` appears inside an `<h1>` tag, you can set the `<h1>` tag’s `id` attribute to `introduction` and omit the anchor link altogether, as shown in the following example:

```
<h1 id="introduction">Introduction</h1>
```

# Choosing Link Colors

---

Links should stand out from the rest of the content in your documents. They need to be easily recognizable by users. Each link has four different status modes:

- **Link** — The standard link in the document. It is neither active nor visited (see other modes).
- **Active** — The link’s target is active in another window.
- **Visited** — The link’s target has been previously visited (typically, this means the target can be found in the user agent’s cache).
- **Hover** — The mouse pointer is over the link.



Each of these modes should be colored differently so that users can tell the status of each link on your page. The usual colors of each link status are as follows:

- **Link** — Blue, underlined text
- **Active** — Red, underlined text
- **Visited** — Purple, underlined text
- **Hover** — No change in the appearance of the link (remains blue, red, or purple)

### Note

As with other presentation attributes in HTML, the user agent plays a significant role in setting link colors and text decorations. Most agents follow the color scheme outlined in this section, but some agents don't conform to this scheme. ■

To change the text color and other link attributes, you can modify the properties of each type of anchor tag. For example, the following style, when used in an HTML document, sets the default visited link text to bold and yellow:

```
a:visited { color: yellow; font-weight: bold; }
```

### Tip

Setting the anchor tag properties without specifying a mode changes all of the link modes to the characteristics of the style. For example, the following style sets all links (link, active, visited) to red:

```
a { color: red; } ■
```

Why would you want to change the color of links in your document? One reason would be that the normal text in your document is the same color as the default link. For example, if your document's text is blue, you probably want to change the default color of existing links from blue to another color to enable users to easily recognize them.

Make a habit of defining all of the link attributes instead of haphazardly defining only one or two of the link status colors. The following styles define each type of link, ensuring that they appear how you want in the document:

```
a:link { color: #003366;
        font-family: verdana, palatino, arial, sans-serif;
        font-size: 10pt; text-decoration: underline; }
a:visited {color: #D53D45;
          font-family: verdana, palatino, arial, sans-serif;
          font-size: 10pt; text-decoration: underline; }
a:active {color: #D53D00;
          font-family: verdana, palatino, arial, sans-serif;
          font-size: 10pt; font-weight: bold;
          text-decoration: underline; }
```

## Part I: Creating Content with HTML

```
a:hover {color: #D53D45;
        font-family:verdana, palatino, arial, sans-serif;
        font-size:10pt; text-decoration: none; }
```

Note the redundancy in the styles — there are only subtle changes in each. You should strive to eliminate such redundancy whenever possible, relying instead upon the cascade effect of styles. You could effectively shorten each style by defining the anchor tag's attributes by itself, and defining only the attributes that are different for each variant:

```
a { color: #003366;
    font-family:verdana, palatino, arial, sans-serif;
    font-size:10pt; text-decoration: underline; }
a:visited {color: #D53D45; }
a:active {color: #D53D00; font-weight: bold; }
a:hover {color: #D53D45; text-decoration: none; }
```

## Link Destination Details

You can add a host of other attributes to your anchor tags to describe the form of the destination being linked to, the relationship between the current document and the destination, and more.

Table 8-2 lists these descriptive attributes and their possible values.

**TABLE 8-2**

**Link Destination Details**

Attribute	Meaning	Value(s)
charset	The character encoding of the target	For example, charset="ISO 8859-1"
hreflang	The base language of the target	For example, hreflang="en-US"
rel	The relationship between the current document and the destination	alternate stylesheet start next prev contents index glossary copyright chapter section subsection appendix help bookmark
rev	The relationship between the destination and the current document	alternate stylesheet start next prev contents index glossary copyright chapter section subsection appendix help bookmark
type	The MIME type of the destination	Any valid MIME type

The following code snippet demonstrates how the relationship attributes (*rel*, *rev*) can be used:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Chapter 10</title>
</head>
<body>
<p><a href="contents.html" rev="chapter" rel="contents">Table of
Contents</a></p>
<p><a href="chapter9.html" rev="next" rel="prev">Chapter 9</a></p>
<p><a href="chapter11.html" rev="prev" rel="next">Chapter 11</a></p>
.....
```

Here, the anchor tags define the relationships between the chapters (next, previous) and the table of contents (chapter, contents).

## The Link Tag

---

You can use the link tag (`<link>`) to provide additional information about a document's relationship to other documents, independently of whether the current document actually links to other documents or not. The link tag supports the same attributes as the anchor tag, but with a slightly different syntax:

- The link tag does not encapsulate any text.
- The link tag does not have an ending tag.

For example, the following code could be used in `chapter10.html` to define that document's relationship to `chapter9.html` and `chapter11.html`:

```
<link href="chapter9.html" rev="next" rel="prev" />
<link href="chapter11.html" rev="prev" rel="next" />
```

The link tag does not result in any visible text being rendered, but it can be used by user agents to provide additional navigation or other user-interface tools.

Another important use of the link tag is to provide alternate content for search engines. For example, the following link references a French version of the current document (`chapter10.html`):

```
<link lang="fr" rel="alternate" hreflang="fr"
    href="chapter10-fr.html" />
```

Other relationship attribute values (*start*, *contents*, and so on) can likewise be used to provide relevant information about document relationships to search engines.

### Summary

---

This chapter covered links — what they are and how to use them to reference other content on the Web. You learned how to construct a link and what attributes are available to the anchor and link tags. You also learned how to define relationships between your document and other documents, and why this is important.

From here, you should progress through the next few chapters, continuing to familiarize yourself with the other various pieces of an HTML document, such as tables, frames, and forms (Chapters 9 through 11). This part of the book then covers colors and multimedia (Chapters 12 and 13), and special characters and internationalization (Chapters 14 and 15), and wraps up with the basics of coding in HTML.

# Tables

**T**ables are a powerful HTML tool with many uses. Developed originally to help communicate tabular data (usually scientific or academic-based data), tables are now used for many purposes — from simply holding tabular data to the layout of entire pages. This chapter covers the basics of tables and then progresses into more complex uses of this versatile HTML structure.

## Parts of an HTML Table

An HTML table is made up of the following parts:

- Rows
- Columns
- Header cells
- Body cells
- Caption
- Header row(s)
- Body row(s)
- Footer row(s)

Figure 9-1 shows an example of an HTML table, with the various parts labeled. The table is defined by the following code:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
```

### IN THIS CHAPTER

Parts of an HTML Table

Table Width and Alignment

Cell Spacing and Padding

Borders and Rules

Rows

Cells

Table Captions

Row Groups — Header, Body,  
and Footer

Background Colors

Spanning Columns and Rows

Grouping Columns

Formatting with Tables

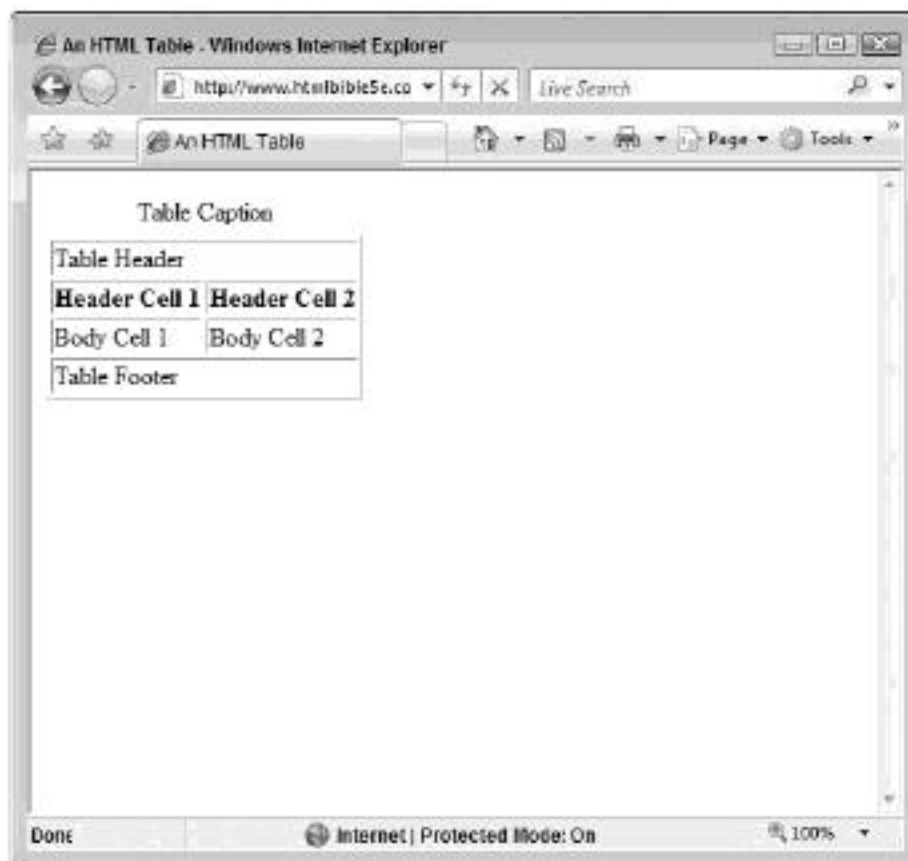
## Part I: Creating Content with HTML

---

```
<title>An HTML Table</title>
</head>
<body>
<p>
  <table border="1">
    <caption>Table Caption</caption>
    <thead>
      <tr><td colspan="2">Table Header</td></tr>
    </thead>
    <tfoot>
      <tr><td colspan="2">Table Footer</td></tr>
    </tfoot>
    <tbody>
      <tr><th>Header Cell 1</th><th>Header Cell 2</th></tr>
      <tr><td>Body Cell 1</td><td>Body Cell 2</td></tr>
    </tbody>
  </table>
</p>
</body>
</html>
```

**FIGURE 9-1**

HTML table elements



Many parts of the HTML table are optional — you need only to delimit the table (with `<table>` tags) and define rows (via `<tr>` tags) and columns (via `<td>` tags). For example, code for a table with these minimum requirements would resemble the following:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
  <title>A Minimal HTML Table</title>
</head>
<body>
<p>
  <table border="1">
    <tr>
      <td>Body Cell 1</td>
      <td>Body Cell 2</td>
    </tr>
  </table>
</p>
</body>
</html>
```

### Tip

It is possible to nest tables within one another. In fact, using tables for layout (covered later in this chapter) depends on this capability. Tables must be nested within table cells (`<td>` tags). See the “Cells” section later in this chapter for more information on the `<td>` tag. ■

## Table Width and Alignment

---

Typically, an HTML table expands to accommodate the contents of its cells. For example, consider the following code and the resulting tables shown in Figure 9-2:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
  <title>HTML Table Widths</title>
</head>
<body>
<p>
  Short Text Table<br />
  <table border="1">
    <tr><td>Short Text 1</td><td>Short Text 2</td></tr>
  </table>
</p>
<p>
  Longer Text Table<br />
```

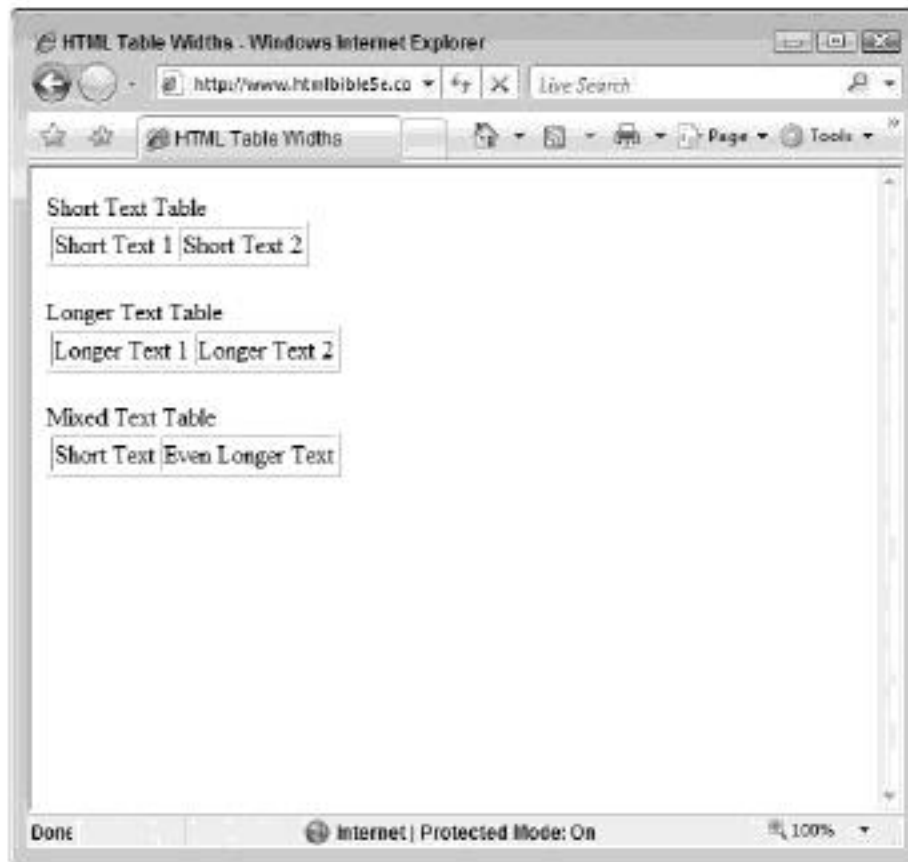
## Part I: Creating Content with HTML

---

```
<table border="1">
  <tr><td>Longer Text 1</td><td>Longer Text 2</td></tr>
</table>
</p>
<p>
  Mixed Text Table<br />
  <table border="1">
    <tr><td>Short Text</td><td>Even Longer Text</td></tr>
  </table>
</p>
</body>
</html>
```

**FIGURE 9-2**

HTML tables expand to accommodate their content.



Once a table expands to the limits of its container object — whether the browser window, another table, or a sized frame — the contents of the cells will wrap, as shown in Figure 9-3.

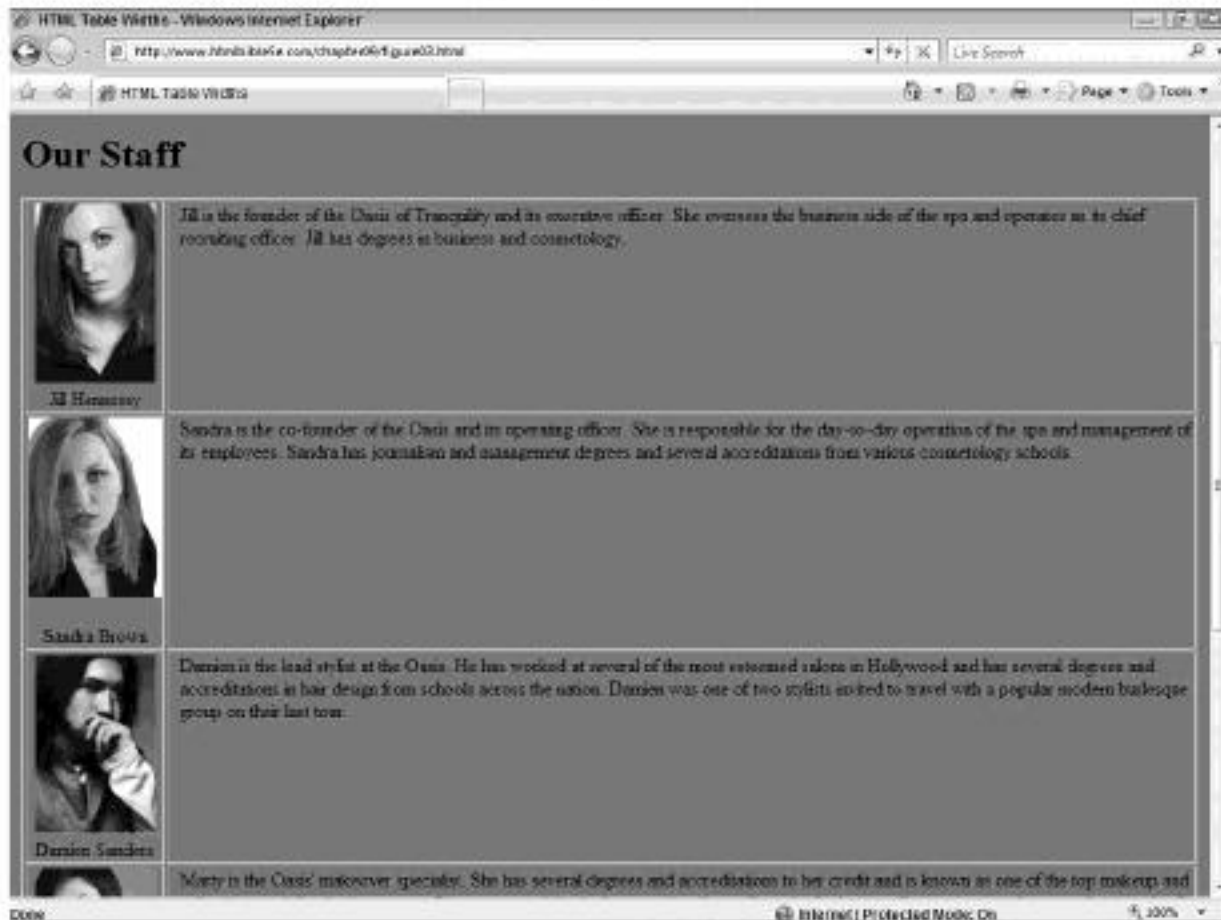
Sometimes you will want to manually size a table, either to fill a larger space or to constrain the table's size. Using the width attribute in the table tag (<table>), you can set a table's



size by specifying the table width in absolute pixels or a percentage of the table's containing object.

**FIGURE 9-3**

Cell contents wrap if a table cannot expand any further.



For example, if you specify 50%, as in the following code snippet, the table's width will be 50 percent of the containing object (which is the width of the browser), as shown in Figure 9-4:

```
<table border="1" width="50%">
```

## Note

Besides specifying the width of the full table, you can also specify the width of each column within the table using width attributes in the table header (<th>) and table cell (<td>) tags. These techniques are covered in the "Cells" and "Grouping Columns" sections later in this chapter. ■

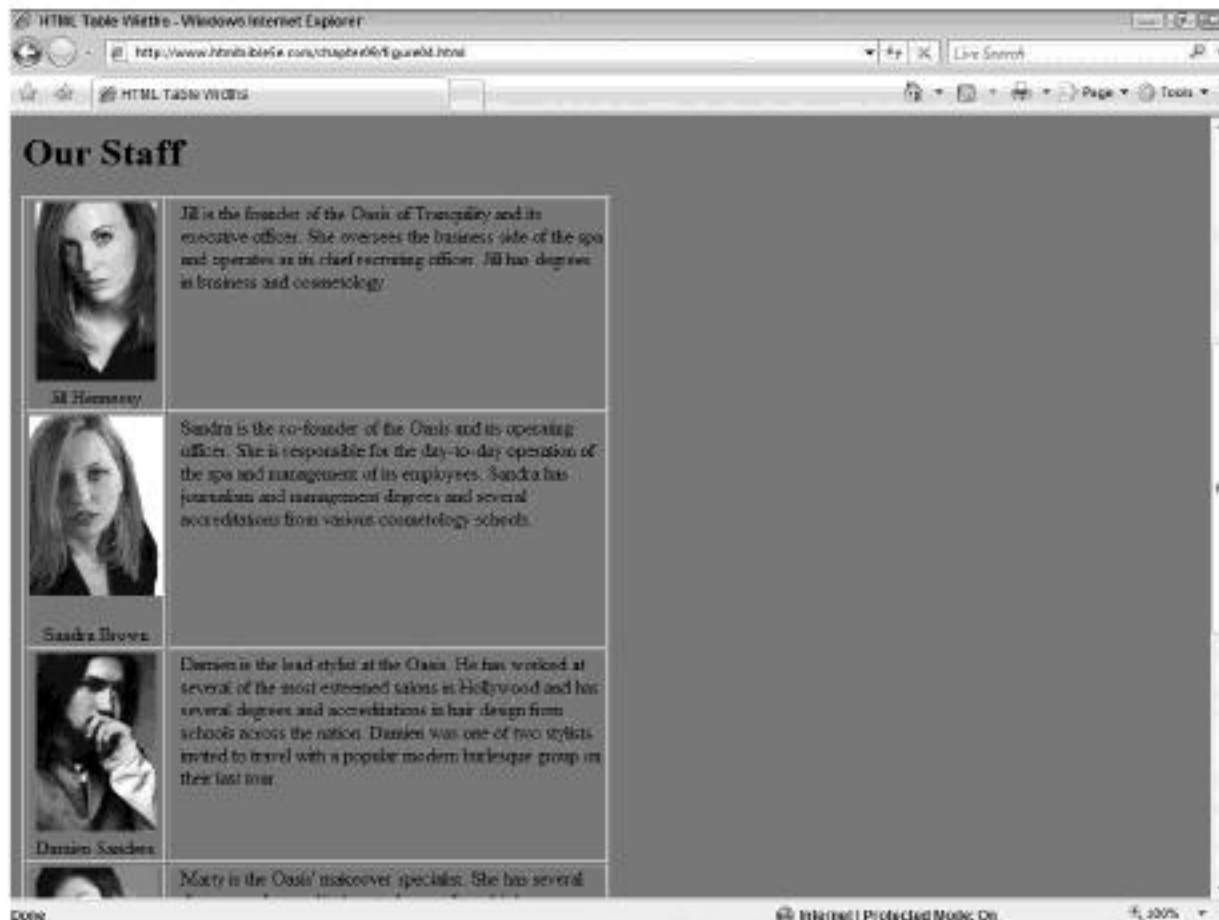
Using a percentage width enables the table to size itself dynamically to the size of its container. For example, if a table is set to 50%, the table will display as 50 percent of its container — paragraph tag, division, or other block object. Note that if the container is not the

## Part I: Creating Content with HTML

width of the user agent, then the scaled table width will not be proportional to the user agent window, but rather the container.

**FIGURE 9-4**

The width of this table is set to occupy 50 percent of the available width of its containing object — in this case, the user agent window.



If you need to specify the exact width of a table, you should specify the width of the table in pixels instead. For example, if you need a table to be 400 pixels wide, you would specify the table with the following tag:

```
<table width="400px">
```

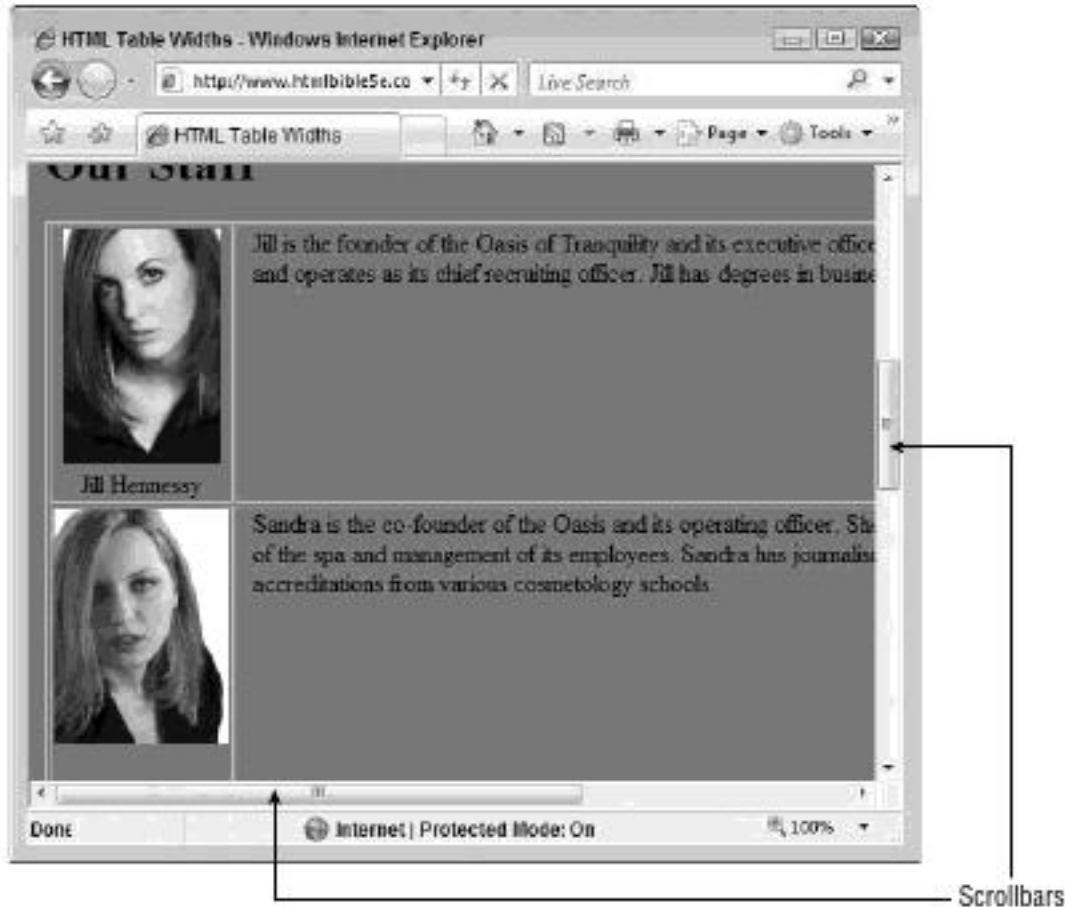
Keep in mind that if the specified table width exceeds the width of its container object, the width will be adjusted to the size of the container. An exception is containers that support horizontal scroll bars; if the container supports scroll bars, then the table will be sized as instructed and the container will spawn scroll bars to accommodate its full width, as shown in Figure 9-5.

### Note

If the table's specified width exceeds the container's width and the container is not scroll bar-enabled, it is up to the browser to handle the table. Most browsers will resize the table to fit the width of its container. ■

**FIGURE 9-5**

Tables too wide for their environment can get some help from scroll bars.



## Cell Spacing and Padding

You can control two cell spacing options in your HTML tables: spacing and padding. *Cell spacing* is the space between cells. *Cell padding* is the space between the cell border and its contents. Figure 9-6 shows the relationship between the two and the cell data itself.

Cell spacing is controlled with the `cellspacing` attribute and can be specified in pixels or percentages. When specified by percentage, the browser uses half of the specified percentage for each side of the cell.

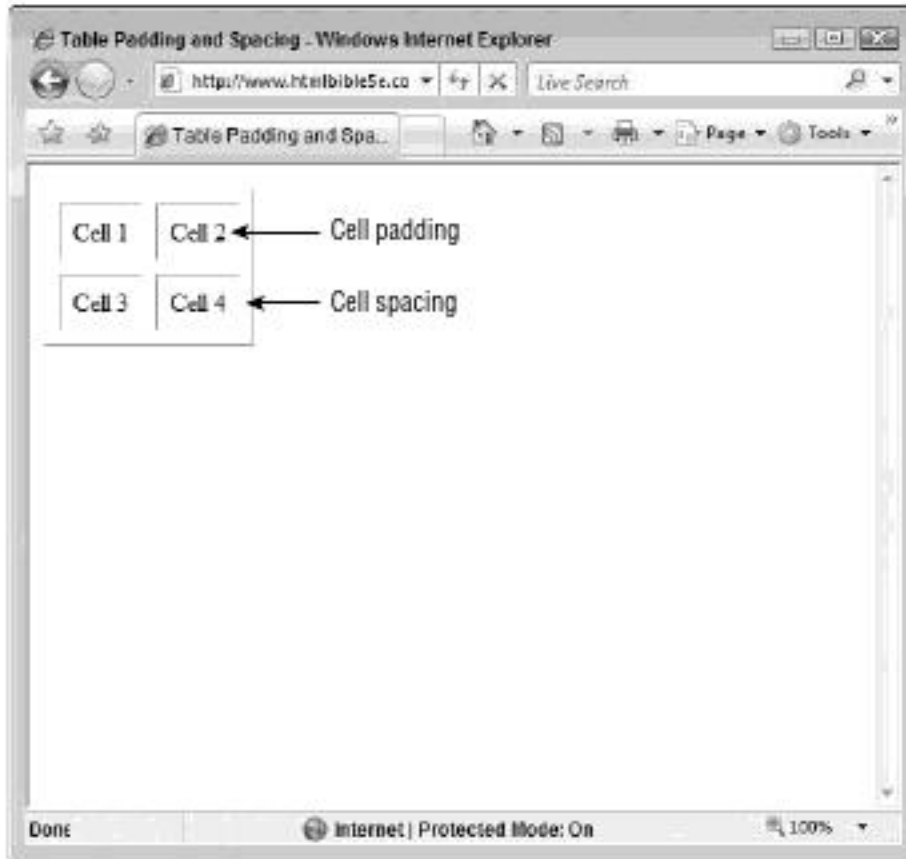
Cell padding is controlled with the `cellpadding` attribute. As with cell spacing, you can specify padding in pixels or a percentage.

### Tip

Keep in mind that cell spacing and cell padding can have a drastic effect on the available size for cell content. Increasing both spacing and padding decreases the cell content size. ■

**FIGURE 9-6**

Cell padding and spacing



## Borders and Rules

---

The border around HTML tables and in between cells can be configured in many ways. The following sections cover the various ways you can configure table borders and rules.

### Table borders

You can use the `border` attribute of the table tag (`<table>`) to configure the outside border of the table. For example, consider the following code containing three tables (the resulting output is shown in Figure 9-7):

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
    <title>Table Outside Borders</title>
</head>
<body>
<p>
```

```

No Borders<br />
<table border="0">
  <tr><td>Cell 1</td><td>Cell 2</td></tr>
  <tr><td>Cell 3</td><td>Cell 4</td></tr>
</table>
</p>
<p>
Border = 1<br />
<table border="1">
  <tr><td>Cell 1</td><td>Cell 2</td></tr>
  <tr><td>Cell 3</td><td>Cell 4</td></tr>
</table>
</p>
<p>
Border = 5<br />
<table border="5">
  <tr><td>Cell 1</td><td>Cell 2</td></tr>
  <tr><td>Cell 3</td><td>Cell 4</td></tr>
</table>
</p>
</body>
</html>

```

**FIGURE 9-7**

Examples of table border widths



## Part I: Creating Content with HTML

The `border` attribute's value specifies the width of the border in pixels. The default border width is 0, or no border.

### Tip

Borders are an effective troubleshooting tool when dealing with table problems in HTML. If you are having trouble determining what is causing a problem in a table, try turning on the borders to better visualize the individual rows and columns. If you are using nested tables, turn on the borders of tables individually until you narrow down the scope of the problem. ■

To specify which outside borders are displayed, use the `frame` attribute with one of the values displayed in Table 9-1.

### Note

Not all user agents follow the defaults for table borders (no borders, or `box`/`border` when a border width is specified). If you want a table to appear with particular formatting, take care to specify all appropriate options, or use CSS to style the table elements. (Table-based CSS properties are covered in Chapter 30.) ■

**TABLE 9-1**

### frame Attribute Values

Value	Definition
<code>void</code>	Display no borders.
<code>above</code>	Display a border on the top of the table only.
<code>below</code>	Display a border on the bottom of the table only.
<code>hsides</code>	Display borders on the horizontal sides (top and bottom) only.
<code>lhs</code> or <code>rhs</code>	Display only the left side or the right side border only.
<code>vsides</code>	Display borders on the vertical sides (right and left) only.
<code>box</code> or <code>border</code>	Display borders on all sides of the table (the default when the <code>border</code> attribute is set without specifying <code>frame</code> ).

## Table rules

You can use the table tag's `rules` attribute to control what rules (borders between cells) are displayed in a table. Table 9-2 shows the `rules` attribute's possible values.

Note that the width of rules is governed by the table spacing attribute. For example, setting `cellspacing` to a value of 5px results in rules five pixels wide.

TABLE 9-2

## rules Attribute Values

Value	Definition
none	Display no rules.
groups	Display rules between row groups and column groups only.
rows	Display rules between rows only.
cols	Display rules between columns only.
all	Rules will appear between all rows and columns.

## Rows

Table rows are the horizontal elements of the table grid and are delimited with table row tags (`<tr>`). For example, a table with five rows would use the following pseudocode:

```
<table>
  <tr> row 1 </tr>
  <tr> row 2 </tr>
  <tr> row 3 </tr>
  <tr> row 4 </tr>
  <tr> row 5 </tr>
</table>
```

The rows are divided into individual cells by embedded `<td>` or `<th>` tags (see the next section, “Cells,” for more details).

### Note

The table row ending tag (`</tr>`) is mandatory. ■

The table row tag supports the attributes shown in Table 9-3.

For an example of how baseline vertical alignment differs from bottom alignment, consider the two tables in Figure 9-8.

If you use the alignment attributes in a `<tr>` tag, that alignment will be applied to all cells in that row. To format cell alignment individually, specify the alignment attribute(s) in individual cell tags (`<th>` or `<td>`) or in `<col>` or `<colgroup>` tags.

### Note

The `bgcolor` attribute, used to set the background color for the row, has been deprecated in HTML 4.01. Instead of using this attribute, I recommend using applicable styles to accomplish the same effect. ■

**TABLE 9-3**

### Table Row Tag Attributes

Attribute	Definition
<code>align</code>	Set to <code>right</code> , <code>left</code> , <code>center</code> , <code>justify</code> , or <code>char</code> , this attribute controls the horizontal alignment of data in the row. Note that if you use <code>char</code> alignment, you should also specify the alignment character with the <code>char</code> attribute described below.
<code>char</code>	Specifies the alignment character to use with character ( <code>char</code> ) alignment.
<code>charoff</code>	Specifies the offset from the alignment character to align the data on. Can be specified in pixels or percentage.
<code>valign</code>	Set to <code>top</code> , <code>middle</code> , <code>bottom</code> , or <code>baseline</code> , this attribute controls the vertical alignment of data in the row. Baseline vertical alignment aligns the text baseline across the cells in the row.

**FIGURE 9-8**

Baseline alignment aligns the baseline of the text.

Bottom Alignment



Baseline Alignment



## Cells

---

The individual cells of a table are the elements that actually hold data. In HTML, cell definitions also define the columns for the table. You delimit cells/columns with table data tags (`<td>`).

For example, consider the following code:

```
<table border="1" cellpadding="5">
  <tr>
    <td>Column 1</td><td>Column 2</td><td>Column 3</td>
  </tr>
  <tr>
    <td>Column 1</td><td>Column 2</td><td>Column 3</td>
```



```

    </tr>
  </table>

```

## Tip

Formatting your tables with ample white space (line breaks and indents) will help you accurately format and understand your tables. There are just as many ways to format a table in HTML as there are Web programmers — find a style that suits your taste and use it consistently. ■

The preceding code defines a table with two rows and three columns, as evidenced by the three sets of `<td>` tags.

You can also use table header tags (`<th>`) to define columns that are headers for the columns. Expanding on the previous example, the following adds column headers:

```

<table border="1" cellpadding="5">
  <tr>
    <th>Header 1</th><th>Header 2</th><th>Header 3</th>
  </tr>
  <tr>
    <td>Column 1</td><td>Column 2</td><td>Column 3</td>
  </tr>
  <tr>
    <td>Column 1</td><td>Column 2</td><td>Column 3</td>
  </tr>
</table>

```

Table header tags make it easy to format column headings without having to resort to character formatting. For example, the preceding code results in most user agents rendering the table header cells in a bold font (the default for `<th>`). To accomplish the same formatting without header tags, you would need to include bold character formatting similar to the following:

```

<tr>
  <td><b>Header 1</b></td>
  <td><b>Header 2</b></td>
  <td><b>Header 3</b></td>
</tr>

```

Using CSS, your formatting options with `<th>` are practically limitless; simply define appropriate formatting or several formatting classes, as necessary.

## Note

Most user agents will not properly render an empty cell (for example, `<td></td>`). When you find yourself needing an empty cell, get in the habit of placing a nonbreaking space entity ( ) in the cell (for example, `<td> </td>`) to ensure that the user agent renders your table correctly. Technically, this “fix” should not be necessary — setting the `empty-cells` style property to `show` should ensure that empty cells are rendered as such. ■

Although cells represent the smallest element in a table, surprisingly, they have the most attributes for their tags. Supported attributes include those shown in Table 9-4.

**TABLE 9-4**

### Cell Attributes

Attribute	Definition
<code>abbr</code>	An abbreviated form of the cell's contents. User agents can use the abbreviation where appropriate (indicating a short form of the contents, displaying on a small device, and so on). As such, the value of the <code>abbr</code> attribute should be as short and concise as possible.
<code>align</code>	The horizontal alignment of the cell's contents — <code>left</code> , <code>center</code> , <code>right</code> , <code>justify</code> , or <code>char</code> (character).
<code>axis</code>	Used to define a conceptual category for the cell, which can be used to place the cell's contents into dimensional space. How the categories are used (if at all) is up to the individual user agent.
<code>char</code>	The character used to align the cell's content if the alignment is set to <code>char</code> .
<code>charoff</code>	The offset from the alignment character to use when aligning the cell content by character.
<code>colspan</code>	How many columns the cell should span (default = 1). See the section "Spanning Columns and Rows" for more information.
<code>headers</code>	A space-separated list of header cell <code>id</code> attributes that corresponds with the cells used as headers for the current cell. User agents use this information at their discretion — a verbal agent might read the contents of all header cells before the current cell's content.
<code>rowspan</code>	How many rows the cell should span (default = 1). See the section "Spanning Columns and Rows" for more information.
<code>scope</code>	The scope of the current cell's contents when used as a header — <code>row</code> , <code>col</code> (column), <code>rowgroup</code> , <code>colgroup</code> (column group). If set, the cell's contents are treated as a header for the corresponding element(s).
<code>valign</code>	The vertical alignment of the cell's contents — <code>top</code> , <code>middle</code> , <code>bottom</code> , or <code>baseline</code> .

### Note

Previous versions of HTML also supported a `nowrap` attribute to control whether a cell's contents wrapped or not. In HTML 4.01, this attribute has been deprecated in favor of styles. See Chapters 30 and 32 for more information on styles pertaining to tables and table cells. ■

## Table Captions

Table captions (`<caption>`) provide an easy method to add descriptive text to a table. For example, suppose you wanted to caption a table detailing the benefits of certain membership levels. The following code adds an appropriate caption to a table whose output is shown in Figure 9-9:

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Table Captions</title>
</head>
<body>

    <table width="400" border="1">
        <caption>The Benefits of Membership</caption>
        <tr>
            <th>Service</th>
            <th>Silver</th>
            <th>Gold</th>
        </tr>
        <tr>
            <td>Valet Parking</td>
            <td>&nbsp;</td>
            <td align="center">X</td>
        </tr>
        <tr>
            <td>Manicure Guarantee</td>
            <td align="center">X</td>
            <td align="center">X</td>
        </tr>
        <tr>
            <td>Monthly Makeover</td>
            <td>&nbsp;</td>
            <td align="center">X</td>
        </tr>
        <tr>
            <td>Hair Maintenance</td>
            <td align="center">X</td>
            <td align="center">X</td>
        </tr>
        <tr>
            <td>Massage Discount</td>
            <td align="center">X</td>
            <td align="center">X</td>
        </tr>
        <tr>
            <td>Monthly 30min Massage Included</td>
            <td>&nbsp;</td>
            <td align="center">X</td>
        </tr>
        <tr>
            <td>Light Lunch During Stay</td>
            <td>&nbsp;</td>
            <td align="center">X</td>
        </tr>
    </table>

```

## Part I: Creating Content with HTML

```
<tr>
  <td>Unlimited Tranquility Room Use</td>
  <td align="center">X</td>
  <td align="center">X</td>
</tr>
<tr>
  <td>Unlimited Whirlpool Use</td>
  <td>&nbsp;</td>
  <td align="center">X</td>
</tr>
<tr>
  <td>8 Hour Appointment Guarantee</td>
  <td>&nbsp;</td>
  <td align="center">X</td>
</tr>
</table>

</body>
</html>
```

**FIGURE 9-9**

The table caption, “The Benefits of Membership,” is placed above the table in this example.



Note that the caption tag must appear immediately after the table tag. Captions typically appear centered above the table to which they are attached, although different user agents may interpret the caption differently.

### Cross-Ref

You can use styles to format the caption however you like. For more information on styles, see Part III of this book. ■

## Row Groups — Header, Body, and Footer

---

Simple tables have only one section, the body, which consists of rows and columns. However, you might want to include additional information in your table by defining a table header and footer to complement the information in the body.

For example, the header could contain the header rows, the body could contain the data, and the footer could contain totals for each column. The advantage to breaking up the table into three sections is that some user agents will then allow users to scroll the body of the table separately from the header and footer.

### Note

The HTML 4.01 specification dictates that you must use all three sections — header, body, and footer — if you use any one section. You cannot use only a header section and body section without a footer section, for example. If you don't intend to use one of the elements, you must still include tags for the section, even if the section is otherwise empty. ■

The table header is delimited by `<thead>` tags — otherwise, its content is exactly like any other table section, delimited by `<tr>`, `<td>`, and optionally `<th>` tags. For example, consider the following table header section:

```
<thead>
  <tr>
    <th>Name</th>
    <th>Hire Date</th>
    <th>Title</th>
  </tr>
</thead>
```

Other than being delimited by `<tbody>` tags, the table body is defined and formatted just like any other table element. The table footer is delimited by `<tfoot>` tags and is formatted like the other two sections.

### Tip

Although it seems counterintuitive, you should place the `<tfoot>` section *before* the `<tbody>` section in your code. This enables the user agent to correctly anticipate the footer section and appropriately format the table body section. ■

## Part I: Creating Content with HTML

---

All three section tags support `align` and `valign` attributes for controlling text alignment within the section for which it applies. (The `char` and `charoff` attributes are also supported for `align = "char"`.)

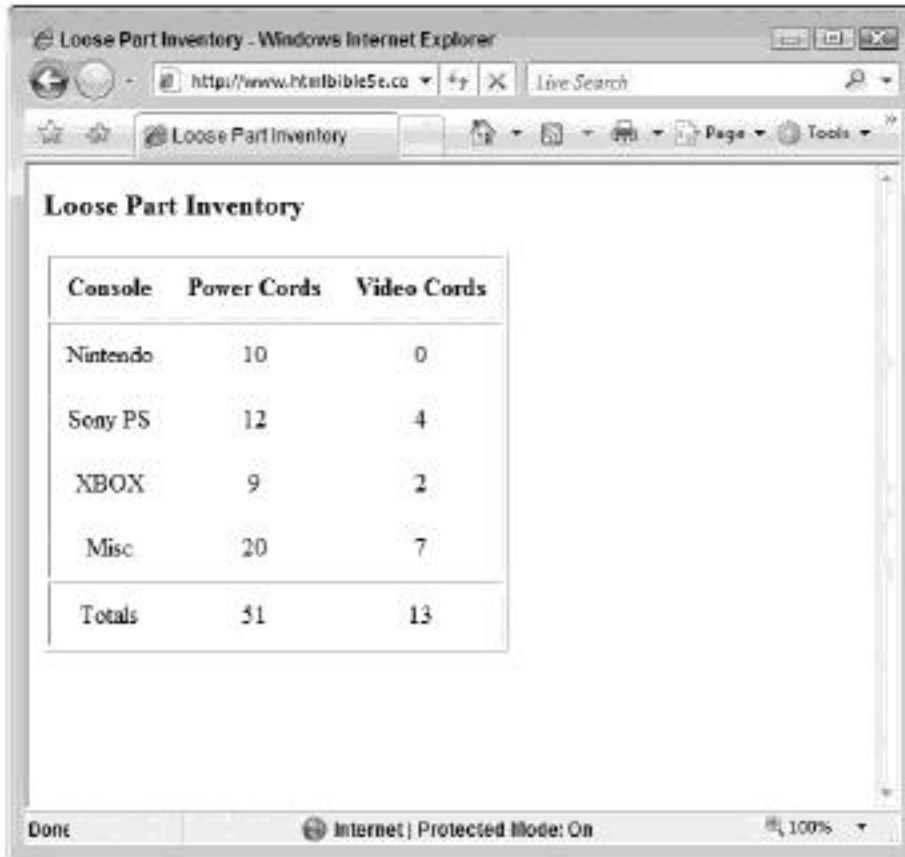
For an example of a table with all three sections, consider the following code and its output, shown in Figure 9-10:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
  <title>Loose Part Inventory</title>
</head>
<body>
<p>
<h3>Loose Part Inventory</h3>
<table border="1" cellpadding="10" cellspacing="2"
  rules="groups">
<thead align="center">
  <tr>
    <th>Controllers</th><th>Power Cords</th><th>Video Cords</th>
  </tr>
</thead>
<tfoot align="center">
  <tr>
    <td>Totals</td><td>51</td><td>13</td>
  </tr>
</tfoot>
<tbody align="center">
  <tr>
    <td>Nintendo</td><td>10</td><td>0</td>
  </tr>
  <tr>
    <td>Sony PS</td><td>12</td><td>4</td>
  </tr>
  <tr>
    <td>XBOX</td><td>9</td><td>2</td>
  </tr>
  <tr>
    <td>Misc</td><td>20</td><td>7</td>
  </tr>
</tbody>
</table>
</p>
</body>
</html>
```

Note how the three sections are set off by rules, but the table is otherwise devoid of rules. This is because of the `rules = "groups"` attribute in the table tag. Also note how alignment attributes are used in the section tags to center the text in the table.

**FIGURE 9-10**

The three table sections (header, body, footer) can be set off by custom rules.



The screenshot shows a web browser window titled 'Loose Part Inventory - Windows Internet Explorer'. The address bar shows 'http://www.htmlbible5e.co'. The page content includes a table titled 'Loose Part Inventory'.

Console	Power Cords	Video Cords
Nintendo	10	0
Sony PS	12	4
XBOX	9	2
Misc	20	7
Totals	51	13

## Background Colors

In previous versions of HTML, you could use the `bgcolor` attribute in the `<table>` and `<tr>`, `<th>`, and `<td>` tags to set a color background for the element. This attribute has been deprecated in HTML 4.01 in favor of using styles to set the background color of table elements.

That said, if you must use the deprecated method, you can set the background of a header row to green with code similar to the following:

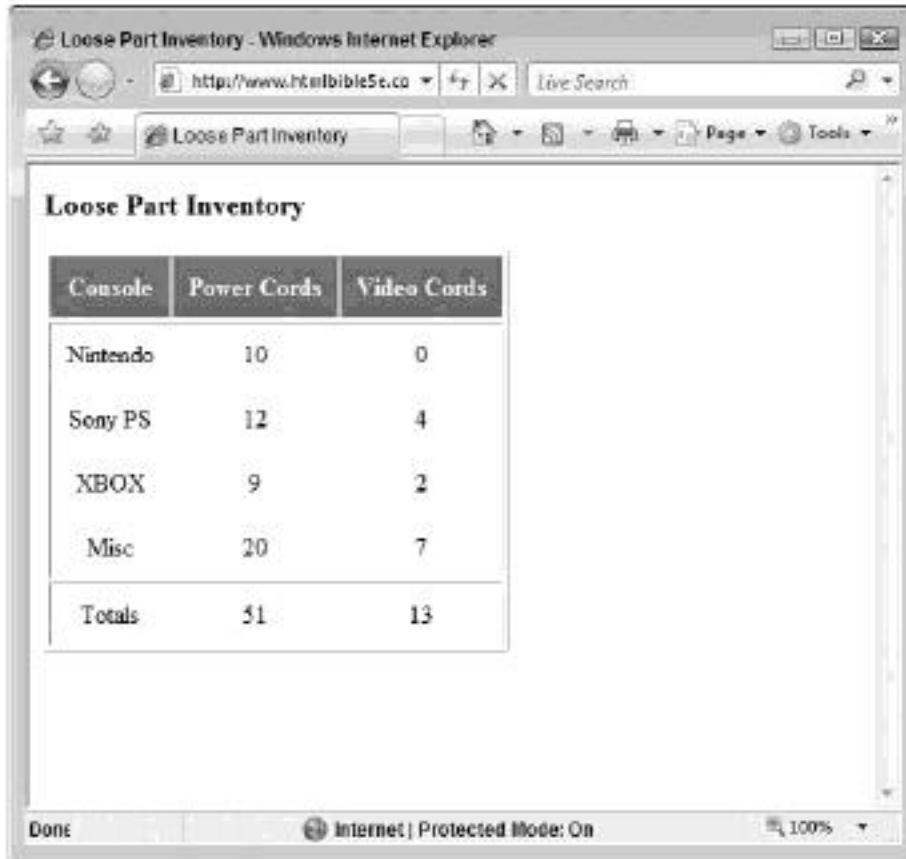
```
<tr bgcolor="green">
  <th>Controllers</th><th>Power Cords</th><th>Video Cords</th>
</tr>
```

If you were to use CSS to accomplish the same effect, the code would resemble the following (output is shown in Figure 9-11):

```
<tr style="background-color: green;">
  <th>Controllers</th><th>Power Cords</th><th>Video Cords</th>
</tr>
```

**FIGURE 9-11**

Use the background-color CSS property to control table element backgrounds.



However, not all user agents adequately support background colors in tables. Older browsers are particularly finicky about correctly representing background colors. When in doubt, test.

## Spanning Columns and Rows

---

It is possible to span data cells across multiple columns and rows using the `colspan` and `rowspan` attributes. Usually such spanning is used to provide column or row headings for groups of columns. For example, consider the following table code utilizing the `colspan` attribute and the resulting output shown in Figure 9-12:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Spanning Columns</title>
</head>
<body>

<table width="400" border="1">
```



```

        <tr>
            <td>&nbsp;</td>
            <td colspan="2">Membership<br />Levels</td>
            <!-- Above cell spans the two membership columns /-->
        </tr>
    <tr>
        <th>Service</th>
        <th>Silver</th>
        <th>Gold</th>
    </tr>
    <tr>
        <td>Valet Parking</td>
        <td>&nbsp;</td>
        <td align="center">X</td>
    </tr>
    <tr>
        <td>Manicure Guarantee</td>
        <td align="center">X</td>
        <td align="center">X</td>
    </tr>
    <tr>
        <td>Monthly Makeover</td>
        <td>&nbsp;</td>
        <td align="center">X</td>
    </tr>
    <tr>
        <td>Hair Maintenance </td>
        <td align="center">X</td>
        <td align="center">X</td>
    </tr>
    <tr>
        <td>Massage Discount </td>
        <td align="center">X</td>
        <td align="center">X</td>
    </tr>
    <tr>
        <td>Monthly 30min Massage Included</td>
        <td>&nbsp;</td>
        <td align="center">X</td>
    </tr>
    <tr>
        <td>Light Lunch During Stay</td>
        <td>&nbsp;</td>
        <td align="center">X</td>
    </tr>
    <tr>
        <td>Unlimited Tranquility Room Use </td>
        <td align="center">X</td>
        <td align="center">X</td>
    </tr>
    <tr>

```

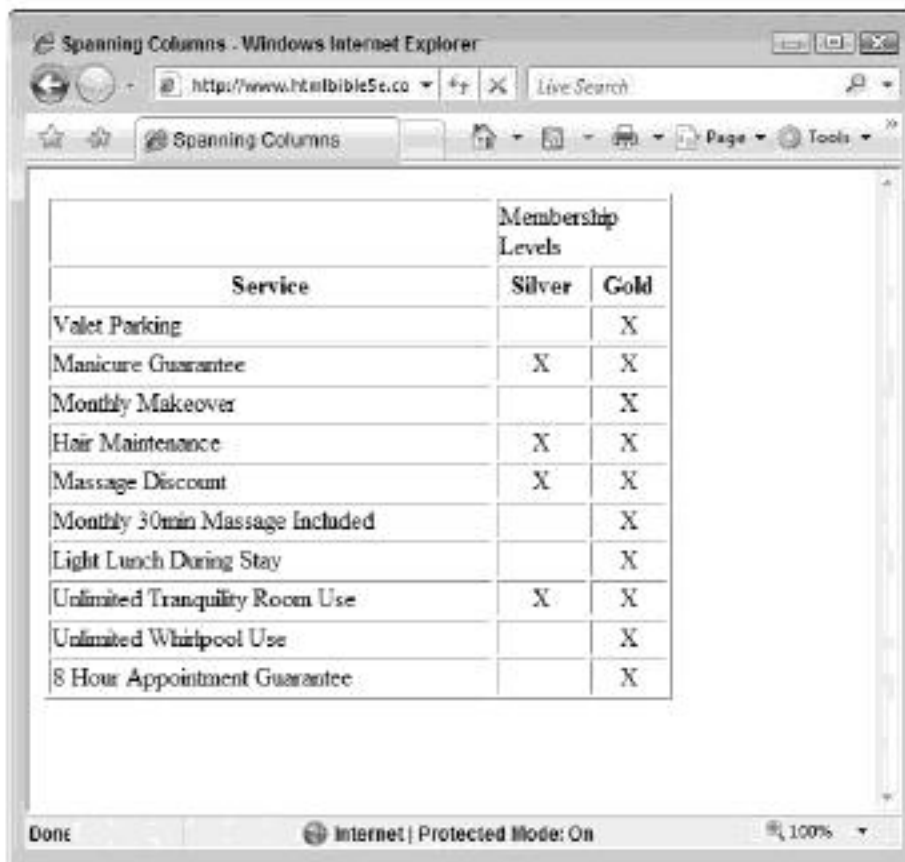
## Part I: Creating Content with HTML

```
<td>Unlimited Whirlpool Use </td>
<td>&nbsp;</td>
<td align="center">X</td>
</tr>
<tr>
<td>8 Hour Appointment Guarantee</td>
<td>&nbsp;</td>
<td align="center">X</td>
</tr>
</table>

</body>
</html>
```

**FIGURE 9-12**

You can span cells across columns.



Service	Membership Levels	
	Silver	Gold
Valet Parking		X
Manicure Guarantee	X	X
Monthly Makeover		X
Hair Maintenance	X	X
Massage Discount	X	X
Monthly 30min Massage Included		X
Light Lunch During Stay		X
Unlimited Tranquility Room Use	X	X
Unlimited Whirlpool Use		X
8 Hour Appointment Guarantee		X

You can span cell rows using the `rowspan` attribute in a similar fashion, as shown in the following code and resulting output in Figure 9-13:

### Note

Rows that include a previously spanned cell omit the declaration of their first cell. ■

**FIGURE 9-13**

Spanning rows with the rowspan attribute

Spanning Rows - Windows Internet Explorer

http://www.htmlbible5e.co

Live Search

Spanning Rows

Premium Services	Service	Silver	Gold
	Valet Parking		X
	Manicure Guarantee	X	X
	Monthly Makeover		X
	Hair Maintenance	X	X
	Massage Discount	X	X
	Monthly 30min Massage Included		X
	Light Lunch During Stay		X
	Unlimited Tranquility Room Use	X	X
	Unlimited Whirlpool Use		X
8 Hour Appointment Guarantee		X	

Done

Internet | Protected Mode: On

100%

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Spanning Rows</title>
</head>
<body>

<table width="400" border="1">
  <colgroup>
    <col></col>
    <col span="2" style="text-align: center;"></col>
  </colgroup>
  <tr>
    <th rowspan="11">Premium<br/>Services</th>
    <!-- Above cell spans 11 rows. Remaining rows omit
         their first cell declaration. /-->
    <th>Service</th>
    <th>Silver</th>
    <th>Gold</th>
  </tr>
```

```
<tr>
  <td>Valet Parking</td>
  <td>&nbsp;</td>
  <td align="center">X</td>
</tr>
<tr>
  <td>Manicure Guarantee</td>
  <td align="center">X</td>
  <td align="center">X</td>
</tr>
<tr>
  <td>Monthly Makeover</td>
  <td>&nbsp;</td>
  <td align="center">X</td>
</tr>
<tr>
  <td>Hair Maintenance</td>
  <td align="center">X</td>
  <td align="center">X</td>
</tr>
<tr>
  <td>Massage Discount</td>
  <td align="center">X</td>
  <td align="center">X</td>
</tr>
<tr>
  <td>Monthly 30min Massage Included</td>
  <td>&nbsp;</td>
  <td align="center">X</td>
</tr>
<tr>
  <td>Light Lunch During Stay</td>
  <td>&nbsp;</td>
  <td align="center">X</td>
</tr>
<tr>
  <td>Unlimited Tranquility Room Use</td>
  <td align="center">X</td>
  <td align="center">X</td>
</tr>
<tr>
  <td>Unlimited Whirlpool Use</td>
  <td>&nbsp;</td>
  <td align="center">X</td>
</tr>
<tr>
  <td>8 Hour Appointment Guarantee</td>
  <td>&nbsp;</td>
  <td align="center">X</td>
</tr>
</table>
```

```
</body>
</html>
```

You can also span columns and rows within the same table by using appropriate `colspan` and `rowspan` attributes. However, such use is not recommended without a GUI HTML editor because the code becomes exponentially complex the more spans you make to a table.

### Cross-Ref

For more information on GUI HTML editors, see Chapter 19. ■

## Grouping Columns

---

HTML 4.01 added a few extra tags to make defining and formatting groups of columns easier. The tags `<colgroup>` and `<col>` are used together to define and optionally format column groups and individual columns, respectively.

The `colgroup` tag is used to define and optionally format groups of columns. The tag supports the same formatting attributes as the `<tr>` and `<td>/<th>` tags (`align`, `valign`, and so on). Any columns defined by the column group tag will inherit the formatting contained within the tag.

To define columns in a group, use the `span` attribute with the `<colgroup>` tag to indicate how many columns are in the group. For example, the following HTML table code places the first three columns in a group:

```
<table>
<colgroup span="3">
</colgroup>
...
```

Note that additional `<colgroup>` tags can be used to create additional column groups. You must use additional column groups if the columns you are grouping are not contiguous or do not start with the first column. For example, the following HTML table code creates three column groups:

- Columns 1 and 2, formatted with centered alignment
- Columns 3–5, formatted with decimal alignment
- Columns 6–10, formatted with right alignment and bold text

```
<table>
<colgroup span="2" align="center">
<!-- This group contains columns 1 & 2 /-->
</colgroup>
<colgroup span="3" align="right" style="text-align: right;">
<!-- This group contains columns 3 - 5 /-->
</colgroup>
```

## Part I: Creating Content with HTML

---

```
<colgroup span="5" align="right" style="font-weight: bold;" >
<!-- This group contains columns 6 - 10 /-->
</colgroup>
...
```

### Note

Column groups that do not have explicit formatting attributes defined in their respective `<colgroup>` tags inherit the standard formatting of columns within the table. However, the group is still defined as a group and will respond accordingly to table attributes that affect groups (rules = "groups", and so on). ■

What if you don't want all the columns within the group formatted identically? For example, in a group of three columns, suppose you wanted the center column (column number 2 in the group) to be formatted with bold text. That's where the `<col>` tag comes into play, defining individual columns within the group. To format a group using the preceding example (middle column bold), you could use code similar to the following:

```
<table>
<colgroup span="3">
<!-- This group contains columns 1 & 3 /-->
<col></col>
<col style="font-weight: bold;"></col>
<col></col>
</colgroup>
...
```

The `<col>` tag follows similar rules to that of the `colgroup` tag — namely, the following:

- Empty tags (those without explicit formatting) are simply placeholders and inherit the formatting of the parent `<colgroup>`.
- You must define columns in order, and in a contiguous group, using blank `<col>` tags where necessary.
- You can use the `span` attribute with a `<col>` tag if you want it to format more than one contiguous column.
- Missing `<col>` tags result in the corresponding columns inheriting the formatting from the parent `colgroup`.

Note that in standard HTML, the column tag has no closing tag. However, in XHTML, the `<col>` tag must be closed by a corresponding `</col>` tag.

### Tip

Column definitions via the `<colgroup>` or `<col>` tags do not eliminate or change the necessity of `td` tags (which actually form the columns). You must still take care in placing your `<td>` tags to ensure proper data positioning within columns. ■

## Formatting with Tables

Formatting your documents with HTML tags enables you to create many useful designs for a variety of purposes. The HTML tag (and related tags) with humble beginnings that revolutionized document formatting with HTML is the table tag (<table>).

The table tag was originally designed to represent tabular data, numbers, and other data in columns. However, using a few tricks, such as embedding tables within one another, it is possible to achieve some pretty fantastic layouts. This section explains how to best utilize tables for page layout purposes.

### Note

With the advent of CSS, there are many who proclaim that tables should no longer be used for any layout purposes, and that instead CSS should be used to style and position elements for the sake of layout. However, this is not necessarily the case. Despite the existence of CSS, HTML tables still make a perfectly acceptable layout mechanism, either on a micro level (such as a simple table of headers and values) or on a macro level (such as the layout basis for an entire page or document).

Arguments can be made for both technologies and the debate can get very heated (try searching for “html table layout versus CSS layout” at [www.google.com](http://www.google.com)). My advice is to use whichever technology makes sense to you — what you are most comfortable with, what presents your documents in the best light, or what appears to be the best tool for the job. ■

## Rudimentary Formatting with Tables

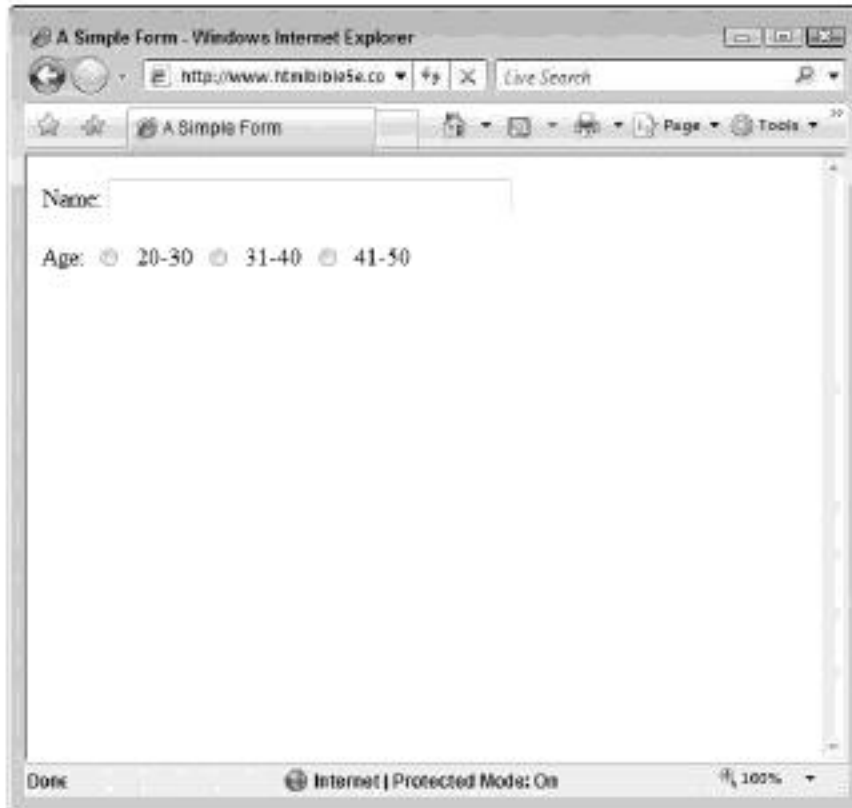
It's not hard to see how tables can help with formatting elements. For example, consider the following code and the output shown in Figure 9-14:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
  <title>A Simple Form</title>
</head>
<body>
<form>
<p>Name:&nbsp;<input type="text" size="40"></p>
<p>Age:&nbsp;<input type="radio" name="20to30" value="20to30">
&nbsp;<input type="radio" name="31to40" value="31to40">
&nbsp;<input type="radio" name="41to50" value="41to50">
&nbsp;</p>
</form>
</body>
</html>
```

## Part I: Creating Content with HTML

**FIGURE 9-14**

A rudimentary form using spaces for layout purposes



A simple table can help better align the elements in this form, as shown in the following code and Figure 9-15:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
    <title>Rudimentary Form Alignment</title>
</head>
<body>
<form>
<table width="50%" border="1">
<tr>
<td width="25%"><p>Name:</p></td>
<td><p><input type="text" size="40"></p></td>
</tr>
<tr>
<td><p>Age:</p></td>
<td><p>
<input type="radio" name="20to30" value="20to30">
&nbsp;20-30&nbsp;
<input type="radio" name="31to40" value="31to40">
&nbsp;31-40&nbsp;

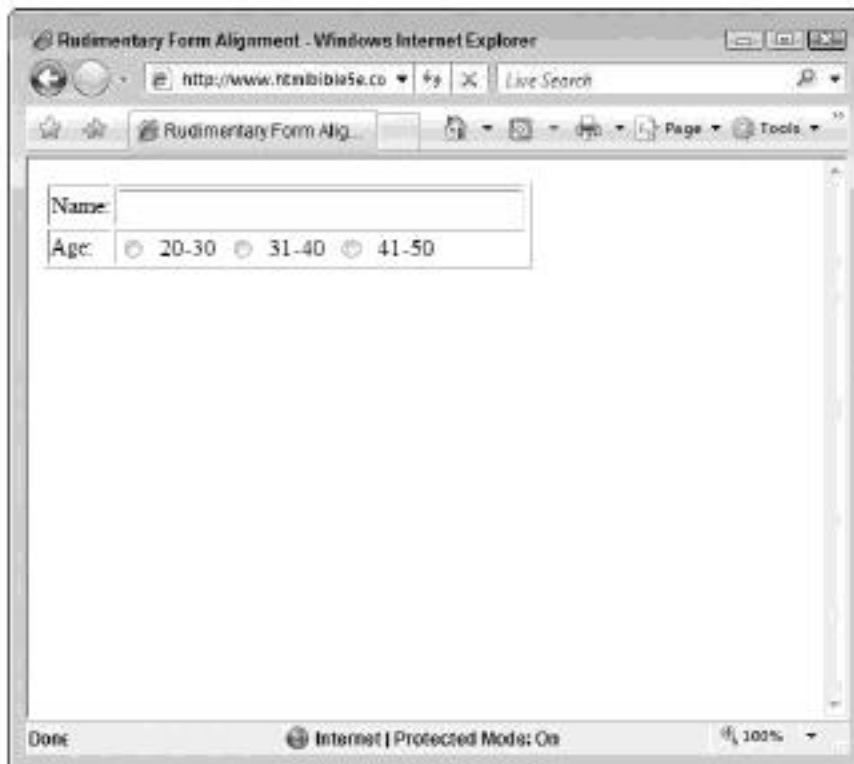
```



```
<input type="radio" name="41to50" value="41to50">
    &nbsp;41-50&nbsp;
</p></td>
</table>
</form>
</body>
</html>
```

**FIGURE 9-15**

Aligning the labels and fields in a form using a simple table



However, this serves only to align the labels and fields in two columns. This is better than no alignment, but if you add a nested table, you can add more order to the radio buttons, as shown in the following code and Figure 9-16:

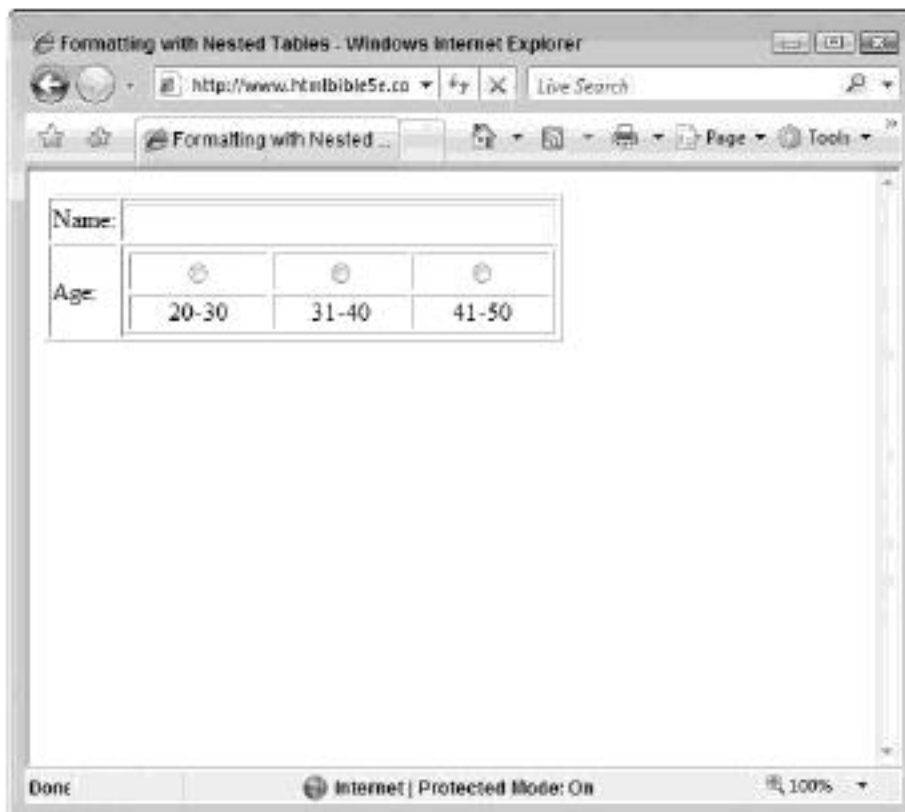
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
    <title>Formatting with Nested Tables</title>
</head>
<body>
<form>
<table width="50%" border="1">
<tr>
<td width="25%"><p>Name:</p></td>
<td><p><input type="text" size="40"></p></td>
```

## Part I: Creating Content with HTML

```
</tr>
<tr>
<td><p>Age:</p></td>
<td>
<table width="100%" border="1">
<colgroup span="3" style="text-align:center;">
</colgroup>
<tr>
<td><p><input type="radio" name="20to30" value="20to30"></p></td>
<td><p><input type="radio" name="31to40" value="31to40"></p></td>
<td><p><input type="radio" name="41to50" value="41to50"></p></td>
</tr>
<tr>
<td><p>20-30</p></td>
<td><p>31-40</p></td>
<td><p>41-50</p></td>
</tr>
</table>
</td>
</table>
</form>
</body>
</html>
```

**FIGURE 9-16**

Nested tables allow for even more alignment and formatting control



## Note

Of course, in real life the tables in the examples would have even more formatting attributes and/or CSS to fine-tune the alignment, and the borders would be off or set to accent the formatting. ■

Even though these examples are fairly small in scope, it should be easy to see the power and flexibility tables can lend to alignment, formatting, and even page layout.

## Real-world examples

You might be surprised by how many tables are hiding under the veneer of the Web pages you frequent. For example, take a look at Figure 9-17, which shows a corporate website.

**FIGURE 9-17**

A corporate website that doesn't visibly use tables

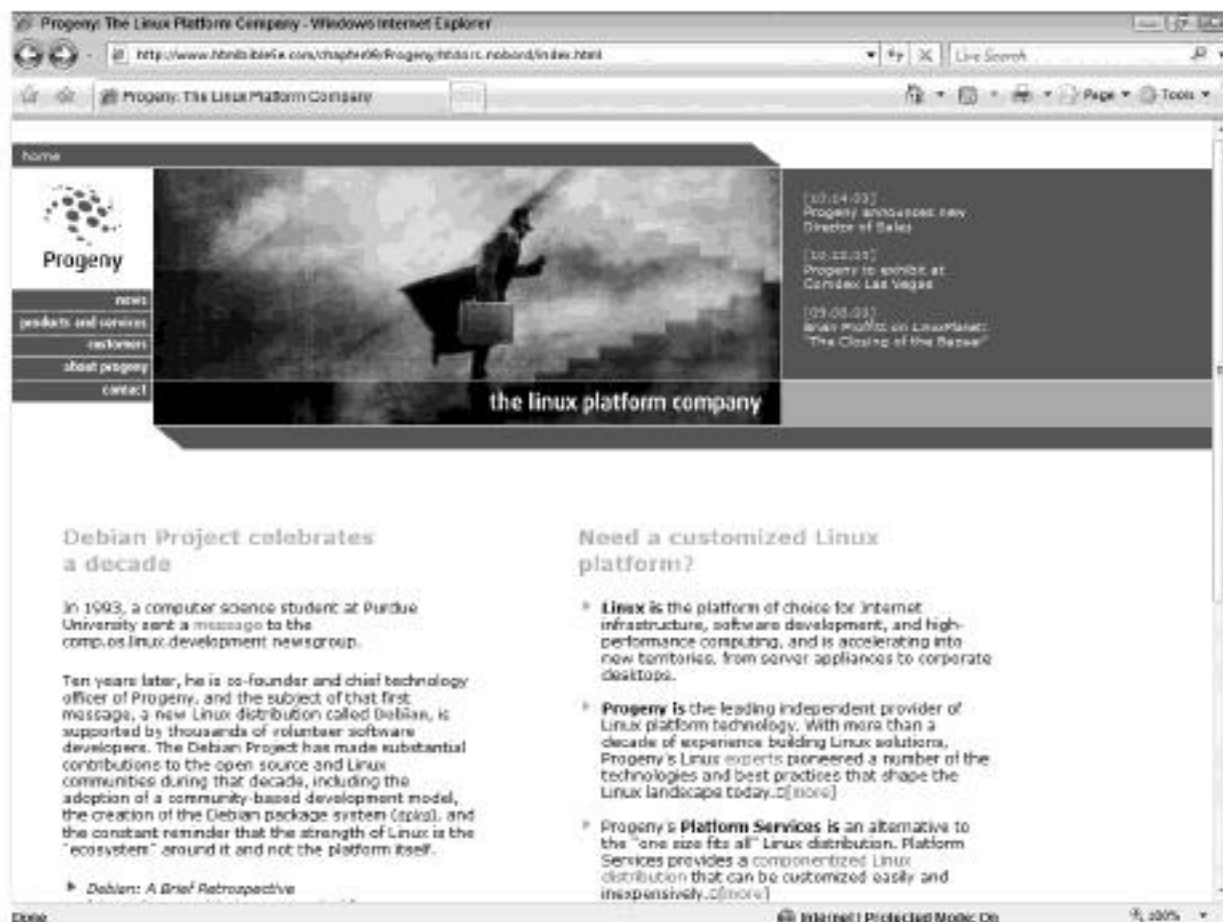


Figure 9-18 shows the same website with the table borders on. Note the multitude of nested tables used to achieve the layout.

## Part I: Creating Content with HTML

**FIGURE 9-18**

A corporate website with the tables made visible



Figure 9-19 shows another popular layout format, a floating page and two columns of content. Again, note that the use of tables, visible in Figure 9-20, isn't readily apparent.

The rest of this chapter shows you how to achieve some of these effects.

### Floating page

The floating page layout has become quite popular and is used in pages of all kinds, from corporate sites to personal Web logs. The effect simulates a piece of paper on a desktop and is fairly easy to create using a few nested tables, as shown in the following code, the output of which is shown in Figure 9-21:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
    <title>Floating Table Format</title>
    <style type="text/css">
        <!-- Sets "desktop" color (behind page) -->
```