

class. To completely understand the link between the two — HTML tags using classes, and styles referencing those classes — you must understand CSS and its methods for accessing class-coded styles. However, keep classes in mind as you code your basic HTML so you can adequately incorporate them.

### Cross-Ref

To get a better idea of how CSS styles work with class-coded HTML tags, see Chapters 25 through 38. ■

Besides linking styles via the `class` attribute, you can also embed specific styles into the individual tag itself using the `style` attribute. The `style` attribute has the following format:

```
style="style-definition; style-definition; style-definition;...
```

This format allows you to specify as many styles as necessary, as long as you separate them with semicolons, as shown in the preceding example.

## Text and Comments

---

Including text that isn't processed by the user agent in your documents can be beneficial for a couple of reasons. The first reason is simple documentation — that is, to make your documents more legible and easy to follow should you need to edit them later. The second is to include more information within the document for later access or inclusion by features supported in upcoming browsers.

### Comments

HTML comments are fairly simple to include in your document, as they have a simple format. A comment in your document might resemble the following:

```
<!-- This is a comment -->
```

Notice that the HTML comment tag is a bit odd, given the dashes and exclamation point in the mix. For clarity, the tag breaks down as follows:

- The tag starts with the standard left angle bracket (<).
- The next character is an exclamation point (!).
- Two hyphens (- -) follow the exclamation point.
- The text of the comment is next.
- The tag starts to close with another set of two dashes (- -).
- The tag closes with a standard right angle bracket (>).

### Note

You cannot nest comments within one another. ■

Comments are best used for short text, not for commenting out large sections of HTML code.

### CDATA sections

Larger comments can make use of CDATA structures — structures created for other markup specifications but enabled in most user agents for XML, as well as HTML (XHTML), rendering. Hence, CDATA is used often for commenting large sections of code in HTML documents.

The format of the CDATA tag is as follows:

```
<![CDATA[    Commented text goes here    ]]>
```

The CDATA tag has vaguely familiar syntax:

- The tag begins with an angle bracket (<).
- The next character is an exclamation point (!).
- The next few characters define the XML tag ([CDATA[).
- The text of the comment is next.
- The tag begins to close with the two brackets (]]).
- The tag closes with the right angle bracket (>).

Like the comment tag, you cannot nest CDATA tags within each other.

#### Tip

The text within a comment or CDATA section is still delivered to the browser and can be seen if the user chooses to reveal the source of the page. The data in these sections is just not rendered as visible text. ■

## Uniform Resource Indicators

---

Uniform resource indicators (URIs) are highly structured lines of text that refer to other resources — locally or on the Internet. In short, URIs are what make the Web the Web — giving pages the ability to provide a link to another page on the Internet. For example, an automobile manufacturer's website may contain links to the different models of cars the manufacturer makes, links to dealerships around the nation, or links to documents of specifications for different vehicle models.

#### Note

The phrase **uniform resource indicator (URI)** is the preferred name for a link on the Internet. Previously, such a link was commonly referred to as a **uniform resource locator (URL)**. ■

The format of the URI is shown in Figure 2-2.

Notice how the URI includes the protocol that should be used to reach the URI resource. This is typically Hypertext Transfer Protocol (HTTP, transferring HTML documents), but it can be other protocols such as File Transfer Protocol (FTP), which transfers all manner of files.

**FIGURE 2-2**

The format of a URI



URIs are used as values for attributes in several different tags, including anchors (used for links to other documents) and images (used to insert images in a document). Consider these two examples, where the URI is underlined for emphasis:

```
<a href="http://www.example.com/detailspecs.html">Details</a>  

```

When you construct a URI for tags in your documents, keep the various pieces of the URI in mind and always try to provide as much detail in your URIs as possible.

### Cross-Ref

For more information on URIs and links, see Chapter 8. ■

## Language and International Options

Several tag attributes can be used to specify language and international options for your documents and individual tags within your documents. The following sections describe those attributes.

### Language code

Most tags support the `lang` attribute, which defines the language in which the content of the tag should be displayed. For example, specifying `en` corresponds to English; `en-US` specifies the United States version of English (as opposed to UK). This attribute has the same format as the rest of the attributes:

```
lang="en-US"
```

### Tip

Valid language codes can be found in RFC1766, a copy of which is online at [www.ietf.org/rfc/rfc1766.txt](http://www.ietf.org/rfc/rfc1766.txt). ■

### Text direction

Along with language specification is text direction. You can specify the direction as right to left (`rtl`) or left to right (`ltr`). The actual direction is specified using the `dir` attribute in whichever tag you want or need to specify it.

### Tip

The Unicode specification, available online at [www.unicode.org/unicode/standard/versions/](http://www.unicode.org/unicode/standard/versions/), provides more details on the direction of text in different languages and conditions. ■

## Summary

---

This chapter covered the basics of supplying attributes to HTML tags, including using proper attribute syntax in your HTML, adding identifiers and class attributes, inserting text and comments in your HTML documents, using a URI, and placing international attributes in your document's tags. Subsequent chapters cover individual tags, their formatting, and specific usage.

# What Goes into a Web Document?

**H**TML has come a long way from its humble beginnings. However, despite the fact that you can use HTML (and its derivatives) for much more than serving up static text documents, the basic organization and structure of the HTML document remains the same.

Before we dive into the specifics of various elements of HTML, it is important to summarize what each element is, what it is used for, and how it affects other elements in the document. This chapter provides a high-level overview of a standard HTML document and its elements. Subsequent chapters cover each element and technology in detail.

## Specifying Document Type

One attribute of HTML documents that is frequently overlooked is the `<!DOCTYPE>` tag, used to specify a Document Type Definition (DTD). This definition precedes any document tags and exists to inform HTML clients of the format of the content that follows — what tags to expect, methods to support, and so forth.

You can think of the DTD as a packing list of sorts that tells the user agent and other clients that read the document what to expect (and not expect) in the document, enabling the client to act more intelligently, anticipating formatting and such. Validation systems use DTDs to actually perform the validation, using the DTD contents as a road map and a syntax guide. HTML editing programs can use the DTD to provide tag auto-completion tools and while-you-type syntax checking.

### IN THIS CHAPTER

**Specifying Document Type**

**Overall Document Structure:  
HTML, Head, and Body**

**Style Definitions**

**Block Elements: Markup for  
Paragraphs and Other Blocks  
of Content**

**Inline Elements: Markup for  
Characters**

**Special Characters (Entities)**

**Organizational Elements**

**Linking to Other Pages**

**Images**

**Comments**

**Scripts**

**Putting It All Together**

The `<!DOCTYPE>` tag is used to specify an existing DTD. It resembles the following:

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

This tag specifies the following information:

- The document's top tag level is HTML (`html`).
- The document adheres to the formal public identifier (FPI) "W3C HTML 4.01 Strict English" standards (`PUBLIC "-//W3C//DTD HTML 4.01//EN"`).
- The full DTD can be found at the URL `www.w3.org/TR/html401/strict.dtd`.

### Note

The DTD concept might be new to even some of the more seasoned Web developers. However, it should be a priority to include an appropriate `<DOCTYPE>` tag in every Web document you produce. Doing so can save you more work in the long run and helps ensure that your documents are rendered as you intend them to be. ■

## Overall Document Structure: HTML, Head, and Body

---

All HTML documents have three document-level tags in common. These tags, `<html>`, `<head>`, and `<body>`, delimit certain sections of the HTML document.

### The `<html>` tag

The `<html>` tag surrounds the entire HTML document. This tag tells the user agent where the document begins and ends. You can think of the `<html>` tag as the virtual top and bottom of your page, as shown in the following:

```
<html>
... document contents ...
</html>
```

Additional language attributes can also be declared within the `<html>` tag. Such options, notably `lang` and `dir` (the language and directional information, respectively) are routinely contained in the document type definition (`<!DOCTYPE>`). However, many experts strongly suggest including the attributes in the `<html>` tag. The `lang` attribute typically takes a two-letter language abbreviation as its value, such as `lang = "en"` for English. The `dir` attribute supports one of two values: `LTR` to specify the text flows left-to-right, or `RTL` to specify the text flows right-to-left.

### The `<head>` tag

The `<head>` tag delimits the HTML document's heading section. The document's title, meta information, and, in most cases, document scripts are all contained in the `<head>` section.



Picture the `<head>` section of the document as the information commonly found in the letterhead or opening section of a printed document. The difference on the Web is that a good portion of the header information is not visible to the end user.

A typical `<head>` section could resemble the following:

```
<head>
<link rel="stylesheet" type="text/css" href="/styles.css" />
<title>On Target Games Home Page</title>
<meta name="description" content="On Target Home Page" />
<meta name="keywords" content="On, Target, Games, Videos" />
<script language="JavaScript">
function NewWindow(url){
fin=window.open(url," ",
"width=800,height=600,scrollbars=yes,resizable=yes");
}
</script>
</head>
```

### Cross-Ref

Most `<head>`-level tags are covered in detail in Chapter 4. JavaScript scripting is covered in more detail in Chapters 16 and 17. ■

The `title` element determines what the user agent displays as the page title. Most user agents display the document title in their title bar, as shown in Figure 3-1.

## The `<body>` tag

The HTML document's main visual content is contained within `<body>` tags. That's not to say that *everything* appearing between the `<body>` tags will be visible, but, like a printed document, this is where the main body of the document is placed and appears.

### Note

With HTML version 4.01, most presentation attributes of the `<body>` tag have been deprecated in favor of specifying these attributes as styles. In previous versions of HTML, you could specify a bevy of options, including the document background, text, and link colors. The `<body>` tag's `onload` and `onunload` attributes, as well as global attributes such as `style`, are still valid. However, you should specify the other attributes in styles instead, as in the following example:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Document Title</title>
<style type="text/css">
body { background: black; color: white}
a:link { color: red }
```

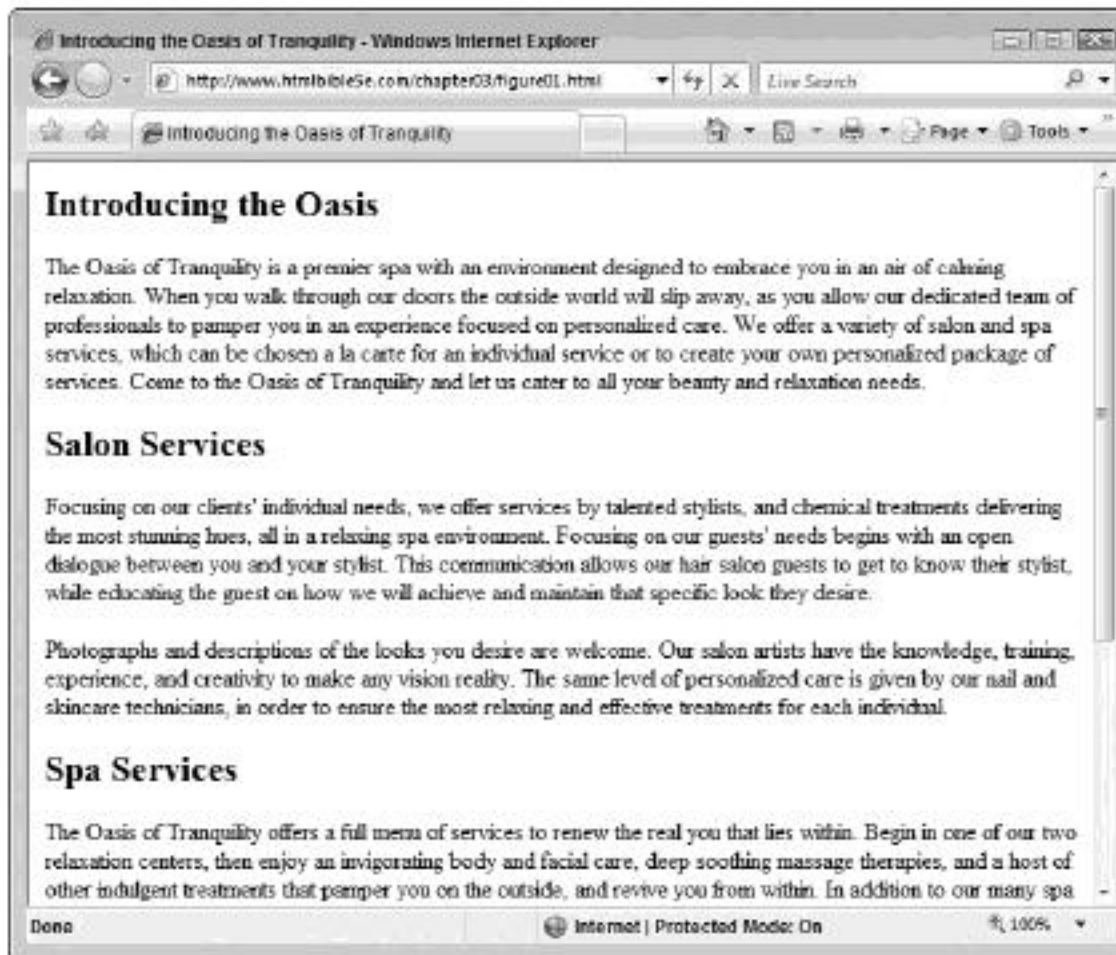
## Part I: Creating Content with HTML

---

```
    a:visited { color: blue; }
    a:active { color: yellow; }
  </style>
</head>
<body>
... document body...
</body>
</html> ■
```

**FIGURE 3-1**

In most user agents, the document's <title> ("Introducing the Oasis of Tranquility") appears in the user agent's title bar.



## Style Definitions

---

Styles have revolutionized how HTML documents are coded and rendered, and they are a technology that should not be neglected when creating Web documents.

Style definitions appear in the head section of a document, linked from a separate file, or are included in individual tags via the `style` attribute.



At their root, styles are simply an aggregation of display attributes combined to achieve a particular result. Those familiar with styles in word processing will have little trouble understanding HTML styles. The important point about styles is that they enable you to radically change a document's appearance by simply applying new styles. This enables you to display the document differently for different uses — different display or output devices, for example — or to provide a different look and feel for different audiences.

It also enables you to make global formatting changes — change one style and every element using that style changes too; there's no need to change every occurrence of the styled element in every document in which it appears.

### Cross-Ref

Styles are covered extensively in the third part of this book, Chapters 25 through 38. ■

## Block Elements: Markup for Paragraphs and Other Blocks of Content

---

As with most word processors, HTML includes several tags to format blocks of text. These tags include the following:

- `<p>` — Formatted paragraphs
- `<h1>` through `<h6>` — Headings
- `<blockquote>` — Quoted text
- `<pre>` — Preformatted text
- `<ul>`, `<ol>`, `<dl>` — Unnumbered, ordered, and definition lists
- `<center>` — Centered text
- `<div>` — A division of the document

It helps to picture each one of these elements formatting paragraph-size chunks of text. Each of the block elements results in a line break and noticeable space padding after the closing tag. As such, the block elements work only on blocks of text — they cannot be used to format characters or words inside blocks of text.

### Cross-Ref

You'll find more details on block elements and their formatting in Chapter 5. ■

## Formatted paragraphs

The paragraph tag (`<p>`) is used to delimit entire paragraphs of text. For example, the following HTML code results in the output shown in Figure 3-2:

```
<p>Welcome to On Target games, the online home of the best-selling  
game, Vanguard Odyssey. Enjoy browsing the site and don't forget  
to check out the updates section.</p>
```

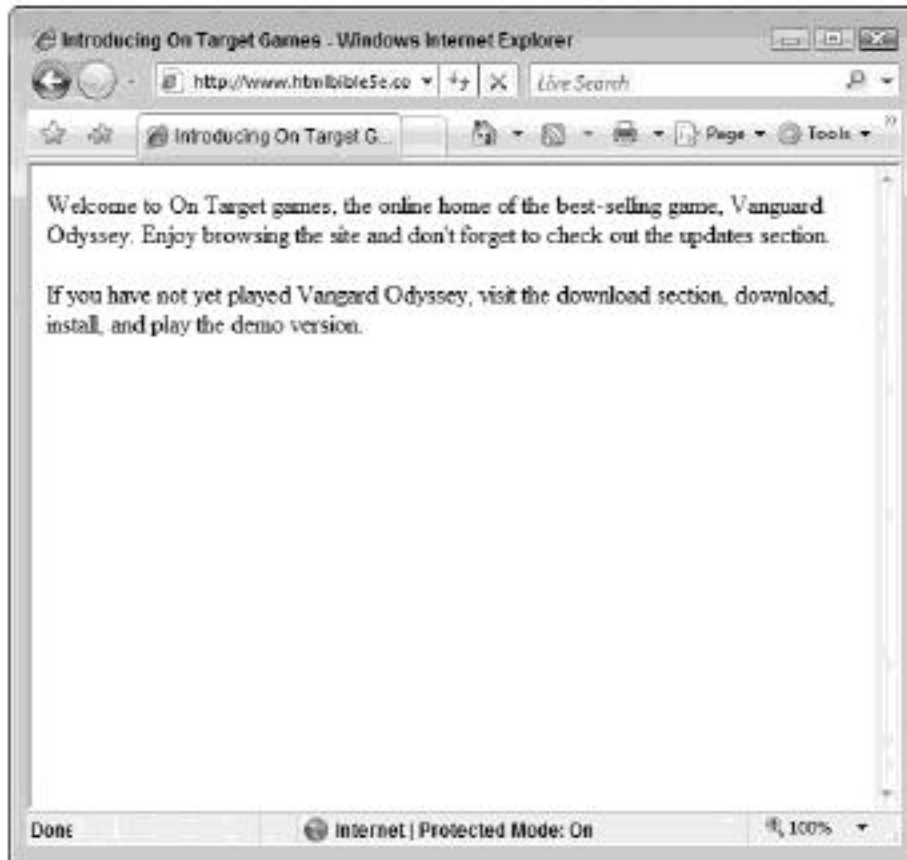
## Part I: Creating Content with HTML

---

```
<p>If you have not yet played Vanguard Odyssey, visit the download  
section, download, install, and play the demo version.</p>
```

**FIGURE 3-2**

Paragraph tags break text into distinct paragraphs.



## Cross-Ref

Paragraph tags are covered in more detail in Chapter 5. ■

## Headings

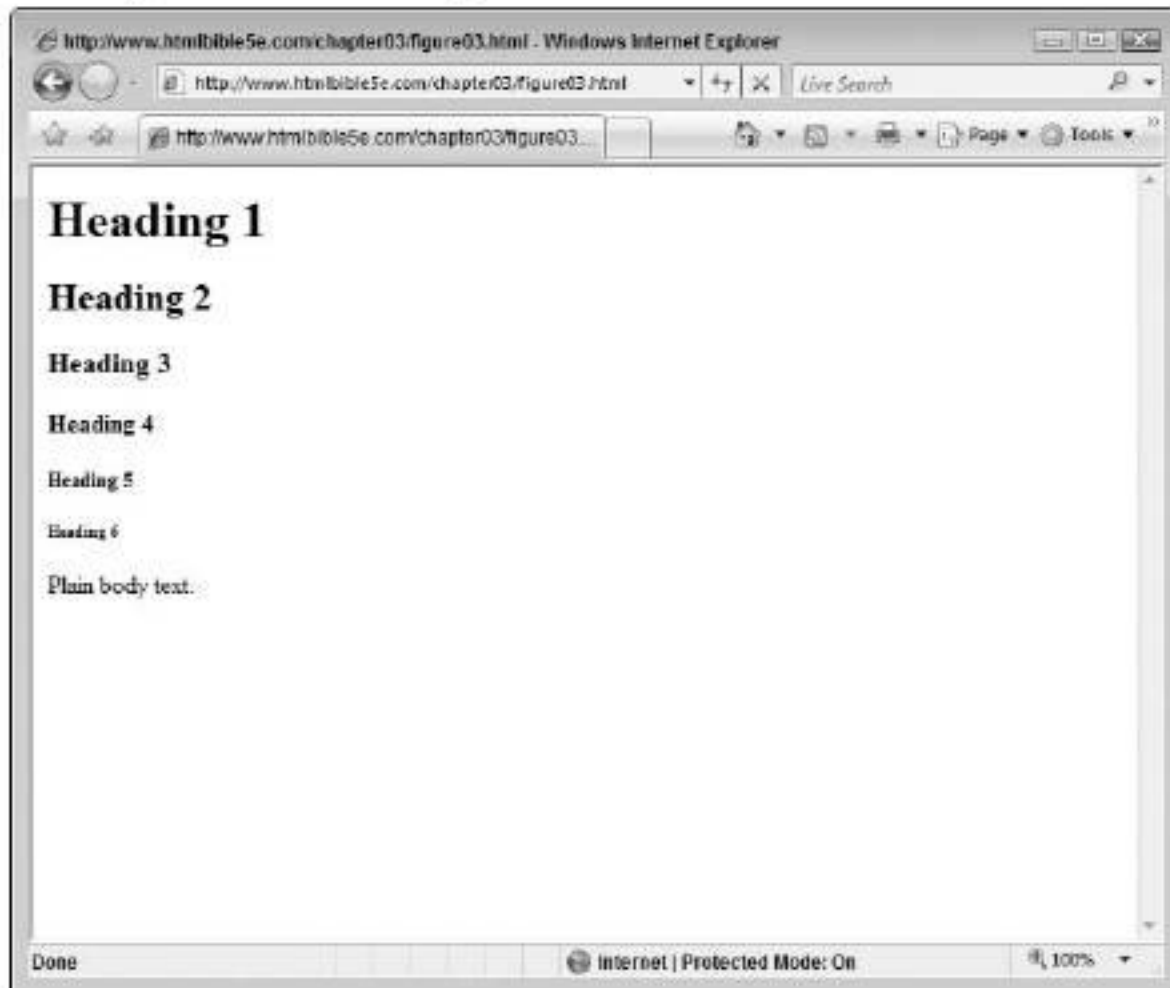
HTML supports six levels of headings. Each heading uses a large, usually bold character-formatting style to identify itself as a heading. The following HTML example produces the output shown in Figure 3-3:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
    "http://www.w3.org/TR/html4/strict.dtd">  
<html>  
  <body>  
    <h1>Heading 1</h1>  
    <h2>Heading 2</h2>  
    <h3>Heading 3</h3>
```

```
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
<p>Plain body text.</p>
</body>
</html>
```

**FIGURE 3-3**

HTML supports six levels of headings.



The six levels begin with Level 1, considered highest or most important, and go to Level 6, the lowest, least important. Although there are six predefined levels of headings, you probably will find yourself using only three or four levels in your documents. There are no restrictions regarding specific levels. You can pick and choose which levels you use; for example, you don't have to use `<h1>` and `<h2>` in order to be able to use `<h3>`. Also, keep in mind that you can tailor the formatting imposed by each level by using styles.

That said, it's a good idea to use the headings the way they were intended — to show the relative importance of one heading to another, and to organize the material. Simply picking a

heading based on its size is a bad idea because you can't always be certain the heading will be rendered in that exact size on every user agent. However, you can be sure the headings will retain their relative size to one another.

### Quoted text

The `<blockquote>` tag delimits blocks of quoted text. For example, the following code sets the review snippet off as a quote:

```
<p>Don't trust us regarding the merits of our game, listen to what  
others have to say:</p>  
<blockquote>  
I'm impressed by the depth of Vanguard Odyssey and its near perfect  
blend of the RPG, shooter, and strategic genres. I give it a hearty  
10 out of 10. - Acme Game Reviews  
</blockquote>
```

The `<blockquote>` tag indents the paragraph to offset it from surrounding text, as shown in Figure 3-4.

### List elements

HTML specifies three different types of lists:

- Ordered lists (usually numbered)
- Unordered lists (usually bulleted)
- Definition lists (list items with integrated definitions)

The ordered and unordered lists both use a list item element (`li`) for each of the items in the list. The definition list has two tags: one for list items (`<dt>`) and another for the definition of the item (`<dd>`).

The following HTML code results in the output shown in Figure 3-5:

```
<ol>To reboot the router, follow these steps:  
  <li>Press and hold the reset button  
  <li>Wait for the power LED to turn red  
  <li>Release the reset button and wait for the power LED to return  
    to green  
</ol>  
<ul>Your new router has these new features:  
  <li>Stateful packet inspection  
  <li>Passthrough VPN support  
  <li>Four gigabit ethernet ports  
</ul>  
<dl>Popular gaming genres:  
  <dt>Action games  
  <dd>Action games are usually "run and gun" games where you run
```

```
around shooting at things. Lots of "action" here.
<dt>Adventure games
<dd>Adventure games are played at a much slower pace. Generally
you follow a storyline that progresses slowly as your character
travels and unravels puzzles.
<dt>Role playing games (RPG)
<dd>Role playing games are very similar to adventure games, except
that the game enforces more investment in your character. The
character advances in capabilities as the game continues allowing
you to "role play" the character instead of simply controlling it.
</dl>
```

**FIGURE 3-4**

The `<blockquote>` tag indents the paragraph.



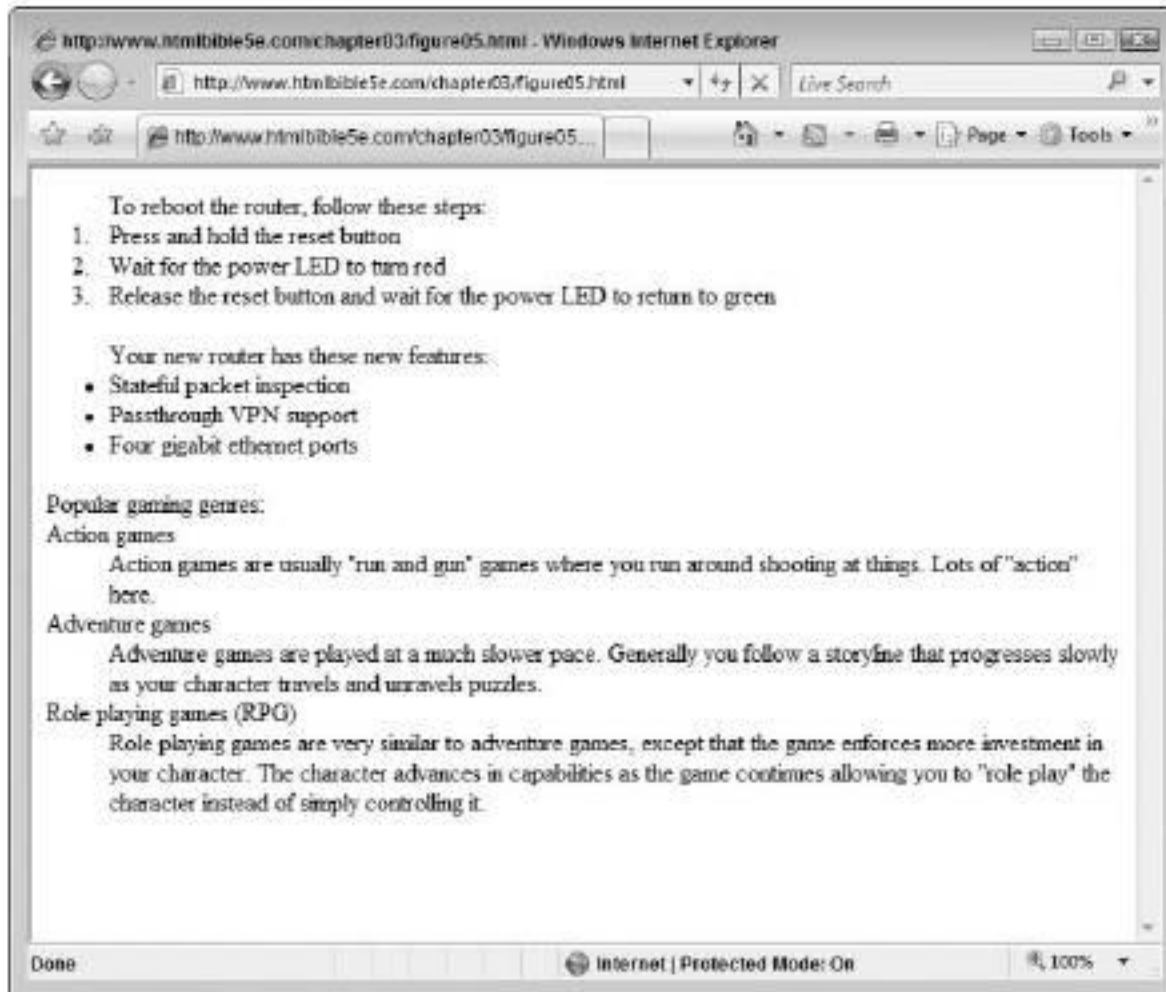
Because of the amount of customization allowed for each type of list, you can create many styles of each list type. For example, you can specify that an ordered list be ordered by letters instead of numbers.

### Cross-Ref

Lists are covered in more detail in Chapter 7. ■

**FIGURE 3-5**

A sample list in HTML



## Preformatted text

Occasionally, you will want to hand-format text in your document or maintain the formatting already present in particular text. Typically, the text comes from another source — cut and pasted into the document — and can be formatted with spaces, tabs, and so on. The preformatted tag (`<pre>`) causes the HTML client to treat white space literally and not condense it as it usually would.

For example, the following table will be rendered just as shown:

```
<pre>
+-----+-----+
| name   | value   |
+-----+-----+
| update | 1069009013 |
| date   | Wed, 8/28, 8:18pm |
+-----+-----+
```



```
| status      | 0 |
| feedupdate | 1069009861 |
+-----+
</pre>
```

### Divisions

Divisions are a higher level of block formatting, usually reserved for groups of related paragraphs, entire pages, or sometimes just a single paragraph. The division tag (<div>) provides a simple solution for formatting sections of a document. Basically, if you need to collect various objects into a larger container, <div> is your tool.

For example, if you need a particular document section outlined with a border, you can define an appropriate style and delimit that part of the document with <div> tags, as in the following example:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Introducing the Oasis of Tranquility</title>
<style>
  .bordered {
    border-style: solid;
    width: 60%;
    padding: 20px;
    margin-left: auto;
    margin-right: auto;}
  .centered {
    text-align: center;}
</style>
</head>
<body>
<h2>Introducing the Oasis</h2>
<p>The Oasis of Tranquility is a premier spa with an environment
designed to embrace you in an air of calming relaxation. When you
walk through our doors the outside world will slip away, as you
allow our dedicated team of professionals to pamper you in an
experience focused on personalized care. We offer a variety of salon
and spa services, which can be chosen a la carte for an individual
service or to create your own personalized package of services.
Come to the Oasis of Tranquility and let us cater to all your beauty
and relaxation needs. </p>
<div class="bordered">
<h2class="centered">Some of Our Specific Services Include</h2>
<h2>Salon Services</h2>
<p>Focusing on our clients' individual needs, we offer services by
talented stylists, and chemical treatments delivering the most
stunning hues, all in a relaxing spa environment. Focusing on our
guests' needs begins with an open dialogue between you and your
```

```
stylist. This communication allows our hair salon guests to get to
know their stylist, while educating the guest on how we will achieve
and maintain that specific look they desire.</p>
<p>Photographs and descriptions of the looks you desire are welcome.
Our salon artists have the knowledge, training, experience, and
creativity to make any vision reality. The same level of
personalized care is given by our nail and skincare technicians, in
order to ensure the most relaxing and effective treatments for each
individual.</p>
<h2>Spa Services</h2>
<p>The Oasis of Tranquility offers a full menu of services to renew
the real you that lies within. Begin in one of our two relaxation
centers, then enjoy an invigorating body and facial care, deep
soothing massage therapies, and a host of other indulgent treatments
that pamper you on the outside, and revive you from within. In
addition to our many spa services, take a refreshing dip in the
swimming pool, melt in one of our whirlpool spas, or rejuvenate
in the sauna. </p>
</div>
<h2>Give the Gift of Tranquility</h2>
<p>All services at the Oasis of Tranquility can be experienced
individually, or selected a la carte to create you own personalized
day of pampering. Gift certificates are excellent for surprising
your loved ones with an hour or a day of pampering and
rejuvenation.</p>
<h2>In Summary...</h2>
<p>So when you are looking for an experience that will relax,
rejuvenate, and free you from the weight and stress of everyday
life and leave you looking and feeling like the person you really
are, come to the Oasis of Tranquility.</p>
</body>
</html>
```

This code results in the output shown in Figure 3-6.

### Cross-Ref

For more information on how to format blocks of text with the `<div>` tag, see Chapter 5. ■

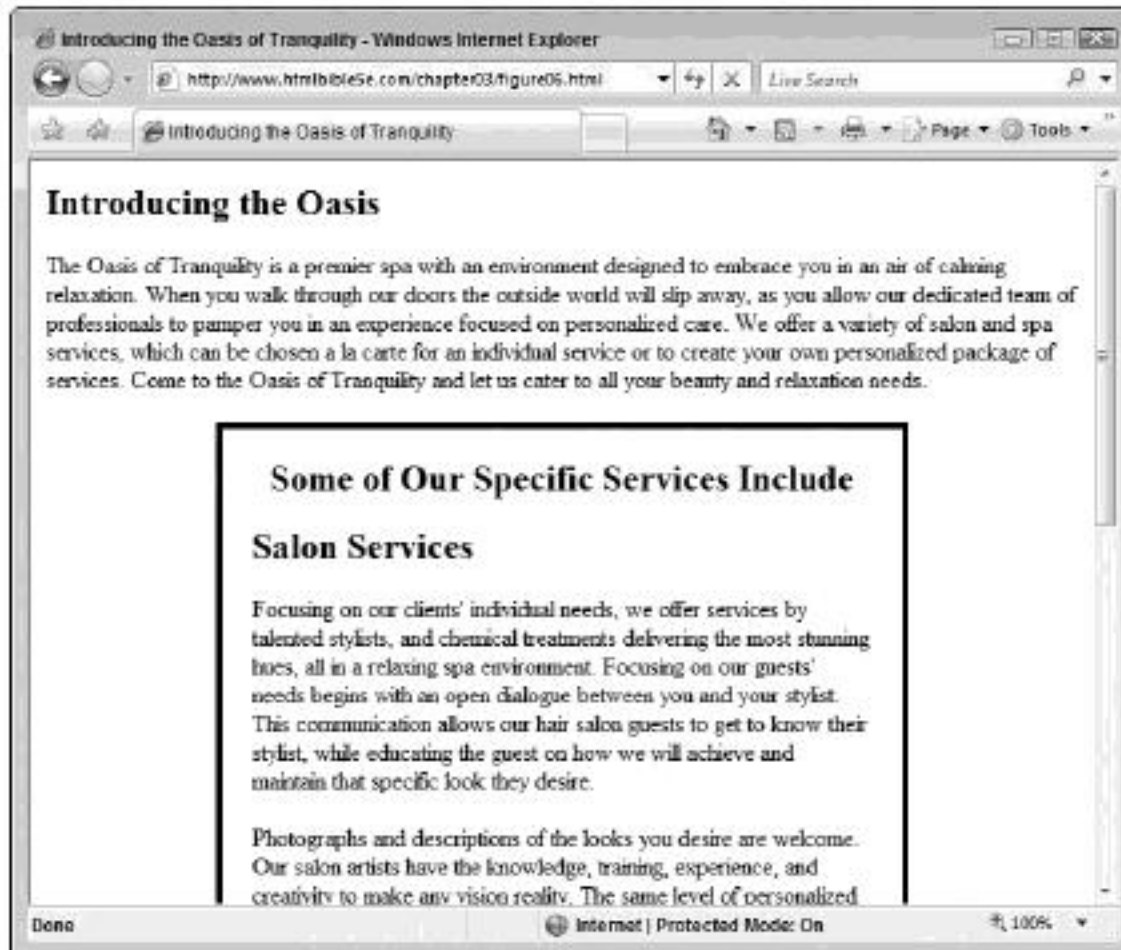
## Inline Elements: Markup for Characters

---

The finest level of markup possible in HTML is at the character level; as in a word processor, you can affect formatting on individual characters. This section covers inline formatting basics.

**FIGURE 3-6**

`<div>` tags delimit sections of text and/or collections of objects.



### Basic inline tags

Inline formatting elements include the following:

- Bold (`b`)
- Italic (`i`)
- Big text (`big`)
- Small text (`small`)
- Emphasized text (`em`)
- Strong text (`strong`)
- Teletype (monospaced) text (`tt`)

## Part I: Creating Content with HTML

---

For example, consider the following sample paragraph, the output of which is shown in Figure 3-7:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<body>
<p>This paragraph shows the various inline styles, such as
<b>bold</b>, <i>italic</i>, <big>big text</big>, <small>small
text</small>, <em>emphasized text</em>, <strong>strong text</strong>,
and <tt>teletype text</tt>.</p>
</body>
</html>
```

**FIGURE 3-7**

Inline elements can affect words or even individual characters.



Note that several inline tags, such as strikethrough (<strike>) and underline (<u>) tags, have been deprecated in the current HTML specifications. Even the font tag (<font>) has been deprecated in favor of styles. As for the strikethrough and underline tags, they have been

replaced by delete (`<del>`) and insert (`<ins>`), which are used for revisions (delete for deleted text, insert for inserted text).

Although it seems counterintuitive, most Web experts recommend using `strong` instead of `bold`, and `emphasized` instead of `italic`, when formatting text. The reasoning has to do with what the styling is supposed to accomplish — strengthen or emphasize text — not how it looks (bold or italic). If you use the appearance styles, most user agents will strive to achieve that particular appearance, even if the representation is different.

### Cross-Ref

Chapter 6 contains more information on inline elements. ■

## Spanning text

Span tags (`<span>`) are used to span styles across one or more inline characters or words. In effect, the `<span>` tag enables you to apply your own inline styles. For example, if you need to specify text that is bold, red, and underlined, you could use code similar to the following:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd"><html>
<head>
<style>
  .emphasis { color: red; text-decoration: underline;
             font-weight: bold;}
</style>
</head>
<body>
<p><span class="emphasis">This text is emphasized as red, bold, and
underlined</span>, while this text is not.</p>
</body>
</html>
```

The `<span>` tag enables you to apply the stylistic formatting inline, exactly where you want it. Without any stylistic additions, the `<span>` tag has no effect on the text it surrounds.

## Special Characters (Entities)

---

Some special characters must be referenced directly instead of simply typed into the document, and some of these characters cannot be typed on a standard keyboard, such as the trademark symbol (™) or the copyright symbol (©). Others could cause the HTML client confusion (such as the angle brackets, `<` and `>`). These specially coded characters are commonly referred to as *character entities*.

## Part I: Creating Content with HTML

---

Entities are referenced by using a particular code in your documents. This code always begins with an ampersand (&) and ends with a semicolon (;). Three different ways to specify an entity exist:

- mnemonic code (such as `copy` for the copyright symbol)
- decimal value corresponding to the character (such as `#169` for the copyright symbol)
- hexadecimal value corresponding to the character (such as `#xA9` for the copyright symbol)

Note that if you use the decimal or hexadecimal methods of specifying entities, you need to prefix the numeric value with a number sign (#).

The following are all examples of valid entities:

- `&nbsp;` — A nonbreaking space (used to keep words together)
- `<` — The less-than symbol, or left-angle bracket (<)
- `&copy;` — The copyright symbol (©)
- `&amp;` — An ampersand (&)
- `&#151;` — An em dash (—)

### Cross-Ref

You'll find more information on entities in Chapter 14. ■

## Organizational Elements

---

Two HTML elements help organize information in a document: tables and forms.

Tables enable you to present data in column and row format, much like a spreadsheet.

Forms enable you to present (and retrieve) data using elements common to GUI interfaces, such as text boxes, check boxes, and lists.

The following sections describe these elements.

### Tables

HTML tables are very basic but can be very powerful when used correctly. At their base level, tables can organize data into rows and columns. At their highest level, tables can provide complicated page design, much like a page in a magazine or newspaper, providing columns for text and sections for graphics, menus, and so on.

Tables have three basic elements and, hence, three basic tags:

- The table definition itself is defined and delimited by `<table>` tags.
- Rows of data are defined and delimited by `<tr>` (table row) tags.
- Table cells (individual pieces of data) are defined and delimited by `<td>` (table data) tags. Alternatively, `<th>` tags can be used for cells in header rows. Table cells, when stacked in even rows, create table columns.



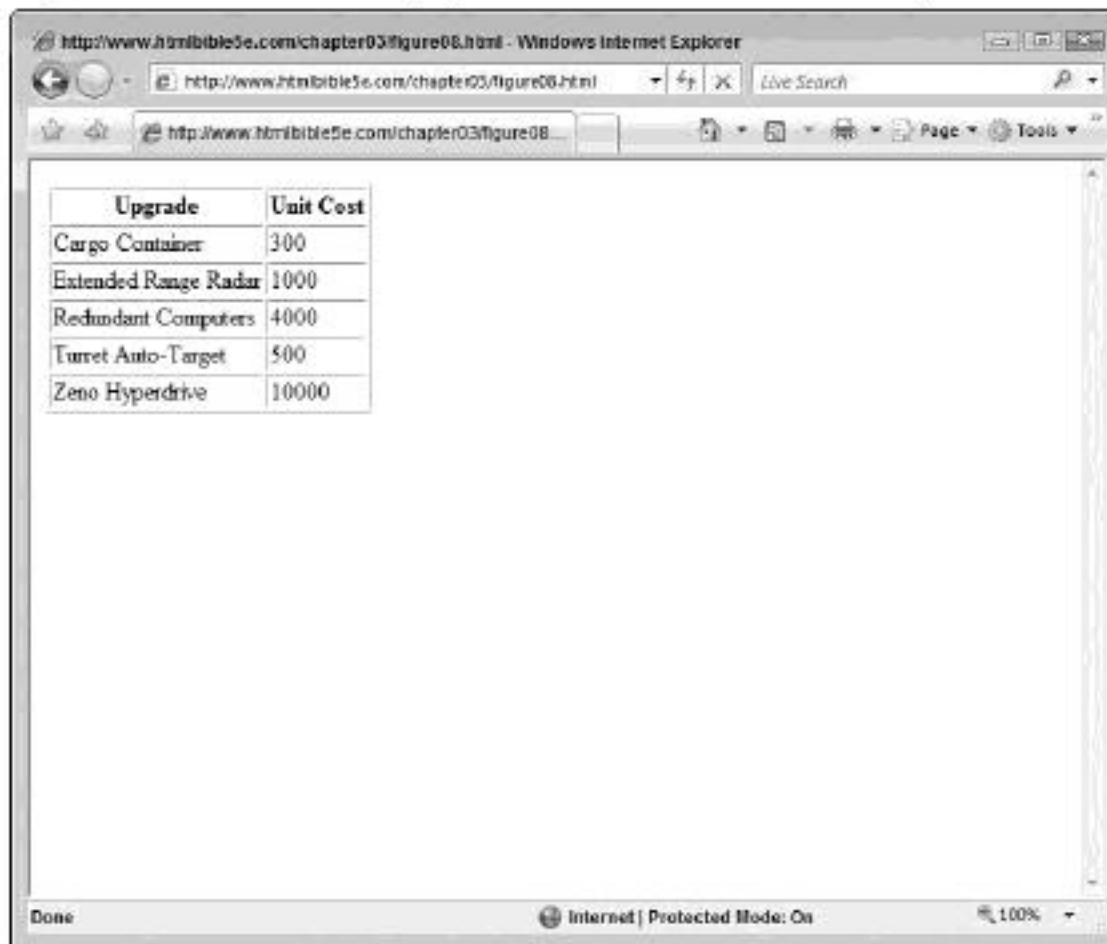
## Chapter 3: What Goes into a Web Document?

For example, consider the following simple table code, which results in the output shown in Figure 3-8:

```
<table border="1">
  <tr><th>Upgrade</th><th>Unit Cost</th></tr>
  <tr><td>Cargo Container</td><td>300</td></tr>
  <tr><td>Extended Range Radar</td><td>1000</td></tr>
  <tr><td>Redundant Computers</td><td>4000</td></tr>
  <tr><td>Turret Auto-Target</td><td>500</td></tr>
  <tr><td>Zeno Hyperdrive</td><td>10000</td></tr>
</table>
```

**FIGURE 3-8**

Simple tables can be used to display data in rows and columns, as in a spreadsheet.



The screenshot shows a web browser window with the address bar displaying <http://www.htmibible5e.com/chapter03/figure08.html>. The browser's title bar reads "http://www.htmibible5e.com/chapter03/figure08.html - Windows Internet Explorer". The main content area displays a table with the following data:

Upgrade	Unit Cost
Cargo Container	300
Extended Range Radar	1000
Redundant Computers	4000
Turret Auto-Target	500
Zeno Hyperdrive	10000

The browser's status bar at the bottom shows "Done", "Internet | Protected Mode: On", and "100%".

This example is straightforward because the table is simple. However, because you can both use a number of options in formatting table elements and nest tables within tables, the tables in your HTML documents can become very complicated (and very powerful).

Tables have long been used to control page layout using similar methods to those used for displaying columnar data. Figure 3-9 shows a Web page that uses tables for its layout; the table borders are displayed to illustrate how the tables enforce the page layout.

**FIGURE 3-9**

Complex tables can be used for complex, custom formatting jobs.



### Cross-Ref

Tables are covered in detail in Chapter 9. ■

## Forms

HTML forms provide a method to use standard GUI elements to display and collect data. HTML forms offer the standard litany of GUI elements, including text boxes, check boxes, pull down (also referred to as drop-down) lists, and more. HTML forms provide a rudimentary method of collecting data and passing that data to a data handler for validation, storage, comparison, or other tasks.

A typical HTML form resembles the following code, the output of which is shown in Figure 3-10:

```
<form>
  <!-- Text field -->
  <b>Name:</b> <input type="text" name="name" size="40">
  <br><br>
```

```
<!-- Radio buttons -->
<b>Age:</b>
  <input type="radio" name="age"> < 20
  <input type="radio" name="age"> 21 -- 30
  <input type="radio" name="age"> 31 -- 40
  <input type="radio" name="age"> 41+
<br><br>
<!-- Select list -->
<b>What is your favorite type of game?</b>
  <select name="game">
    <option name="action">Action
    <option name="adventure">Adventure
    <option name="rpg">RPG
  </select>
<br><br>
<!-- Check boxes -->
<b>How may we contact you for more information?</b><br>
<input type="checkbox" name="phone">Phone<br>
<input type="checkbox" name="mail">Mail<br>
<input type="checkbox" name="email">Email<br>
<input type="checkbox" name="no">Do not contact me<br>
<p><input type="submit" value="Submit" />
<input type="reset" /></p>
</form>
```

### Note

The preceding example form is very simple; it shows only some basic elements, and has no handler to process the data collected by the form. Real-world forms can be quite complex and usually require validation scripts to ensure that the data collected is valid. However, this simple form illustrates the amount of control you can assert over data and format using HTML. ■

### Cross-Ref

Forms are covered in detail in Chapter 11. ■

## Linking to Other Pages

---

The main advantage to the World Wide Web is the ability to link to other documents on the Web. For example, if you had a page that detailed local zoning laws, you might want to include a link to a local government site where additional information could be found. A link typically appears as underlined text and is often rendered in a different color than normal text.

For example, a link might appear in a user agent as follows:

More information can be found [here](#).

The word *here* is linked to the other document; when the user clicks the word, the user agent displays the specified page.

## Part I: Creating Content with HTML

**FIGURE 3-10**

Form elements provide standard GUI controls for displaying and collecting data.



Links are created by use of the anchor tag, `<a>`. At its simplest level, this tag takes one argument — the page to link to — and surrounds the text to be linked. The preceding example could be created with the following code:

```
More information can be found  
<a href="http://www.whitehouse.gov">here</a>
```

The anchor tag's `href`, or Hypertext REference attribute, specifies the protocol and destination of the link. The example specifies `http://` because the destination is a document to be delivered via the HTTP protocol. Other protocols (such as `ftp://` or `mailto:`) can also be used where appropriate.

Additional attributes can be used with the anchor tag to specify such things as where the new document should be opened (for example, in a new window), the relationship between the documents, and the character set used in the linked document.

You can also use a variant of the anchor tag to mark specific places in the current document — a bookmark of sorts. A link can then be placed elsewhere in the document that can take the user to the specific place. For example, consider this HTML code:

```
For more information see <a href="#Chapt2">Chapter 2</a>
. . . More HTML . . .
<a name="Chapt2">Chapter 2</a>
```

In this example, the user can click the Chapter 2 link to move to the location of the Chapter 2 anchor. Note that the href link must include the hash, or pound, symbol (#), which specifies that the link is an anchor instead of a separate page.

### Cross-Ref

More information on links and anchors can be found in Chapter 8. ■

## Images

---

One of the great innovations the World Wide Web and HTTP brought to the Internet was the ability to serve up multimedia to clients. The precursors to full-motion video and CD-quality sound were graphical images in the Web-friendly Graphics Interchange Format (GIF) and Joint Photographic Experts Group (JPEG) format.

You can include images in HTML documents by using the image tag (<img>). The image tag includes a link to the image file as well as pertinent information used to display the image (for example, the image's size). A typical image tag resembles the following:

```

```

The preceding example would result in the image `tmoore.jpg` being displayed at the location in the document where the tag appears. In this case, the image is in the `images` directory of the current server and will be displayed without a border, 100 pixels wide by 200 pixels high. The `alt` attribute provides a textual alternative for the visually impaired, or user agents that cannot display graphics (or whose users have configured them not to). The attribute can also be used to display additional information about the image, as most user agents will show the attribute's value as a tooltip when the mouse is hovered over the image.

Images can also be navigation aids, enabling the user to click certain parts of an image to perform an action, display another document, and so on. For example, a map of the United States could be used to help users select their state — clicking a state would bring up the applicable page for that state. Navigational images are commonly referred to as *image maps* and require a separate map of coordinates and geometric shapes to define the clickable areas.

### Cross-Ref

You'll find more information on images in Chapter 12. ■

### Comments

---

Although HTML documents tend to be fairly legible all on their own, there are several advantages to adding comments to your HTML code. Some typical uses for comments include aiding in document organization and document-specific code choices, or marking particular document sections for later reference.

HTML uses the tag `<!--` to begin a comment, and the tag `-->` to end a comment. Note that the comment can span multiple lines, but the user agent ignores anything between the comment tags. For example, the following two comments will both be ignored by the user agent:

```
<!-- This section needs better organization. -->
and
<!-- The following table needs to include these columns:
    Age
    Marital Status
    Employment Date
-->
```

### Scripts

---

HTML is a static method of deploying content; the content is sent out to a user agent where it is rendered and read, but it typically doesn't change once it is delivered. However, there is a need in HTML documents for such things as decision-making ability, form validation, and, in the case of Dynamic HTML (DHTML), dynamic object attribute changes. In those cases (and more), client-side scripting can be used.

### Cross-Ref

**For more information on client-side scripting, see Chapters 16 and 17. ■**

Client-side scripting languages, such as JavaScript, have their code passed to the user agent inside the HTML document. It is the client's responsibility to interpret the code and act accordingly. Most client-side scripts are contained in the HTML document's `<head>` section within `<script>` tags, similar to the following example:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
    <script language="JavaScript">
        function MiscWindow(w,h,url){
            opts = "width="+w+",height="+h;
            opts += ".scrollbars=no,resizable=yes";
            fin = window.open(url,"",opts);
        }
    </script>
</head>...
```



You can also include the JavaScript in an external file and use the `<script>` tag's `src` attribute to reference it. For example, the following script section references the external script file `utility.js`:

```
<script type="text/JavaScript" src="utility.js"></script>
```

Note that the `<script>` section still includes an opening and closing tag. When would you want to place your code in an external file? When the scripts are used by multiple documents, placing the code in an external file enables you to reference the one copy from multiple documents and to maintain one copy of the code, not one copy per document.

In many cases, the document uses events as triggers to call the script(s). Events can be connected to scripts via HTML event-handler attributes. These attributes can be included in links (`onclick`), forms (`onchange`), and elements such as the body tag (`onload`, `onunload`).

## Putting It All Together

---

As you can see, the standard HTML document is a fairly complex beast. However, when taken piece by piece, the document is really just like any other document.

The following HTML listing shows how all of these pieces fit together:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
  <meta ... meta tags go here ... >
  <title> ... title of the page/document goes here< .../title>
  <link rel="stylesheet" href=" ...external style sheet name ..."
        type="text/css">
  <style>
    ... any document specific styles go here ...
  </style>
  <script>
    ... client-side scripts go here ...
  </script>
</body>
  ... body of document goes here ...
</body>
</html>
```

All HTML documents should have a `<DOCTYPE>` declaration, `<html>` and `<body>` tags, and at least a `<title>` within the `<head>` section. The rest of the elements are strictly optional, but they help define a document's purpose, style, and ultimately its usability, as you will see in subsequent chapters.

### Summary

---

This chapter covered the basic elements that make up a Web document — the frame of the document, the heading, and the basic markup for the content. You learned what a document type definition is and why it's important to specify one for your documents. You read about framing tags `<html>`, `<head>`, and `<body>`, including styles in your documents, tag elements used for marking up blocks of content, and those for marking up inline pieces of text. In addition, this chapter discussed placing character entities in your content, tag elements used to organize content, linking to other documents and sections, as well as including images, comments, and scripts in your documents.

# The HEAD Elements

**Y**ou have seen what various pieces make up HTML documents and how they all fit together. In Chapters 5 through 14 you will see how each individual piece is formatted and placed in the document. This specific chapter deals with the elements in the head section of the document.

## Specifying the Document Title

The `<head>` element of an HTML document contains several other elements, including the document title. The document title is delimited between `<title>` tags and can include any character or entity. For example, consider the following `<head>` section, which includes a registration mark:

```
<title>Welcome to On Target Games &reg;</title>
```

This title shows in the title bar of Internet Explorer, as shown in Figure 4-1.

While it is useful to have the title of your document in the title bar of the client's browser, the title is used in several other locations as well. It is used as the default shortcut/favorite name in most browsers, linked to in most search engines, and so on. As such, you should always include a title for your documents, and make it as descriptive (but concise) as possible.

## Providing Information to Search Engines

Your document's `<head>` section can also include `<meta>` tags. These tags are not rendered as visible text in the document; they are used to pass information and commands to the client browser.

### IN THIS CHAPTER

Specifying the Document Title

Providing Information to  
Search Engines

Setting the Default Path

Script Sections

Style Sections

Specifying Profiles

Background Color and  
Background Images

**FIGURE 4-1**

Entities are rendered correctly in document titles.



As its name implies, the `<meta>` tag contains *meta information* for the document. Meta information is data about the document itself, instead of information about the document's contents. Most of a document's meta information is generated by the Web server that delivers the document. However, using meta tags you can supply different or additional information about the document.

The amount of information you can specify with `<meta>` tags is extensive. If you use the `HTTP-EQUIV` parameter in the `<meta>` tag, you can supply or replace HTTP header information. For example, the following `<meta>` tag defines the content type of the document as HTML with the Latin character set (ISO-8859-1):

```
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
```

In addition, you can control some aspects of how the client browser treats the document. You can specify how long the document should be cached (if cached at all), refresh the browser with a different page after a delay, and so forth. For example, the following two `<meta>` tags tell the browser not to cache the current page (`pragma, no-cache`) and to refresh the browser window with a different page after three seconds (`refresh`):

```
<meta http-equiv="pragma" content="no-cache" />
<meta http-equiv="refresh"
content="3;URL=http://www.example.com/newpage.html" />
```

### Note

For a comprehensive list of HTTP 1.1 headers, see the HTTP 1.1 definition on the W3C website: [www.w3.org/Protocols/rfc2616/rfc2616.html](http://www.w3.org/Protocols/rfc2616/rfc2616.html). ■

A bevy of supplemental data can be included using `<meta>` tags. Many Web authoring tools embed their name, version, and assorted authoring information, for example. In addition, you can include categorization information and assorted short notes.

Most search engines have stopped using the meta description and meta keywords as the sole source of indexing a document, relying instead on their own, proprietary indexing methods. However, several smaller robots still use these fields and the majors sometimes reference them.

The description and keywords data is provided by the following two `<meta>` tags:

```
<meta name="description" content="The affordable day spa" />
<meta name="keywords" content="facial treatments, hair,
manicures, pedicures, relaxation, spa, pools, sauna" />
```

## Setting the Default Path

---

When defining links and references in your HTML document, be as exact as possible with your references. For example, when referencing a graphic with an `<img>` tag, you should make a habit of including the protocol and the full path to the graphic, as shown here:

```

```

However, it isn't very practical to type the full path to every local element referenced in your document. As such, a document residing on the `example.com` server could reference the same graphic with the following code:

```

```

What if the document is relocated? The images directory might no longer be a subdirectory of the directory where the document resides. The image might be on a separate server altogether.

To solve these problems, you could use the `<base>` tag. The `<base>` tag sets the default document base — that is, the default location for the document. Using the preceding example, a document in the root directory of the `example.com` server would have a `<base>` tag similar to the following:

```
<base href="http://www.on-target-games.com/images/" />
```

Any absolute references in the document (those with full protocol and path) will continue to point to their absolute targets. Any relative references (those without full protocol and path) will be referenced against the path in the `<base>` tag. Meta tags can also be used to refresh a document's content or redirect a client browser to another page. Refreshing a document is useful

## Part I: Creating Content with HTML

---

if it includes timely, dynamic data, such as stock prices. Redirection comes in handy when a document moves, as you can use a redirect to automatically send a visitor to the new document.

To refresh or redirect a document, use the `http-equiv="refresh"` option in a `<meta>` tag. This option has the following form:

```
<meta http-equiv="refresh" content="seconds_to_wait; url" />
```

For example, suppose that a page on your site (`on-target-games.com`) has moved. The page used to be on the server's root as `listing.html`, but now the page is in an `oldpage` directory as `listing.html (/oldpage/listing.html)`. You want visitors who previously bookmarked the old page to be able to get to the new page. Placing the following document in the server's root (as `bio.html`) would cause visitors to automatically be redirected to the new page after a three-second wait:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
  <title>This Page Has Moved!</title>
  <meta http-equiv="pragma" content="no-cache" />
  <meta http-equiv="refresh" content="3;
    URL=http://www.on-target-games.com/oldpage/listing.html" />
</head>
<body>
<p>This page has moved. You will be redirected to the new page
in 3 seconds, or you can click the link below.</p>
<a href="http://www.on-target-games.com/oldpage/listing.html" >
The new page.</a>
</body>
</html>
```

To refresh the current page every three seconds, use the following:

```
<meta http-equiv="refresh" content="3" />
```

### Tip

Using the `pragma no-cache` meta tag along with the `refresh` meta tag is always a good idea. This helps keep the browser from caching the document and displaying the cached copy of the document instead of the updated document. Because different browsers treat `no-cache` differently, it is also wise to add an `expires` meta tag, as shown here:

```
<meta http-equiv="expires" content="0" />
```

This tag causes the document to be immediately expired in the cache and, hence, not cached at all. ■

The refresh technique is especially useful on pages that show timely information. By forcing the page to reload at certain intervals, you can help ensure that site visitors see current information.



### Script Sections

---

HTML documents can include scripting sections. Such sections typically contain JavaScript scripts, but other types of scripting (for example, VBScripting) can also be used.

All scripting in a document should appear between `<script>` tags. These `<script>` sections can be placed in the document head section or anywhere in the body. The `<script>` tags should adhere to the following format:

```
<script type="text/javascript">
...scripting code...
</script>
```

Note that the MIME type of the code (in this case `text/javascript`) is included as a `type` attribute.

#### Note

The exception to where code should appear in a document is within event attributes within specific tags. For more information, see Chapters 16 and 17. ■

### Style Sections

---

Style blocks are another large section that can appear in the head section of the document. Style blocks are formatted as shown in the following listing:

```
<style type="text/css">
...style definitions...
</style>
```

#### Note

As with scripts, styles can also be used as attributes of HTML tags (the `style` attribute). More information on styles can be found in Chapters 25 through 38. ■

### Specifying Profiles

---

Profiles are an interesting concept, allowing XML-based data structures to be attached to HTML documents. These profiles enable compatible readers to return the profile information, specifying items such as the document's author, last modification date, and more.

A profile document is a properly formatted HTML document consisting of a definition list containing terms and data of the profile itself. Consider the following document:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
xml:lang="en" lang="en" >
```

```
<head><title>HTML Profile Document</title></head>
<body>
  <dl class="profile">
    <dt id="author">author</dt>
    <dd>Author of the document</dd>
    <dt id="keywords">keywords</dt>
    <dd>A list of the keywords for the document.</dd>
    <dt id="copyright">copyright</dt>
    <dd>Copyright information for the document</dd>
    <dt id="date">date</dt>
    <dd>The date that the document was last updated.</dd>
  </dl>

</body>
</html>
```

Additional information can be added to the profile as additional definitions, as required.

To tie the profile to a document, you use the profile attribute in the document's `<head>` tag. For example, suppose the profile document were at the following address:

```
http://www.example.com/profiles/profile1.html
```

You would use the following `<head>` tag:

```
<head profile="http://www.example.com/profiles/profile1.html">
```

Applicable readers and browsers can use this information to access the profile appropriately.

## Background Color and Background Images

One of the easiest changes you can effect on your Web pages is to alter the background color of your document. Most browsers use a white background, so specifying a different background color or a background image can easily make your document distinct.

### Specifying the document background color

If you code your HTML against the transitional format of HTML, you can use the `bgcolor` attribute in the `<body>` tag. However, using that attribute is not recommended for the following reasons:

- The attribute is not valid for strict HTML and might impair the validation of your document.
- If you want to change your documents' background color, you must change each individual body tag in each document.

A better practice is to use appropriate styles, typically in an external style sheet.

The document background color is set using the `background-color` property. For example, to set the background color to blue, you would use the following style definition:

```
<style type="text/css">
  body { background-color: blue;}
</style>
```

### Cross-Ref

For more information on styles, refer to Chapters 25 through 38. ■

## Specifying the document background image

Besides setting the document's background to a solid color, you can also specify an image to use as the background. As with the background color attribute for the body tag, there is also a background image attribute (`bgimage`) for the body tag. However, as with the background color attribute, it is not a good idea to use that attribute.

Instead, use the `background-image` property as a body style, as shown here:

```
<style>
  body { background-image: url(<char:Variable>path_to_image</char:Variable>);}
</style>
```

For example, the following style results in `grid.jpg` being placed as the document's background:

```
<style type="text/css">
  body { background-image: url(images/grid.jpg);}
</style>
```

The effect is shown in Figure 4-2.

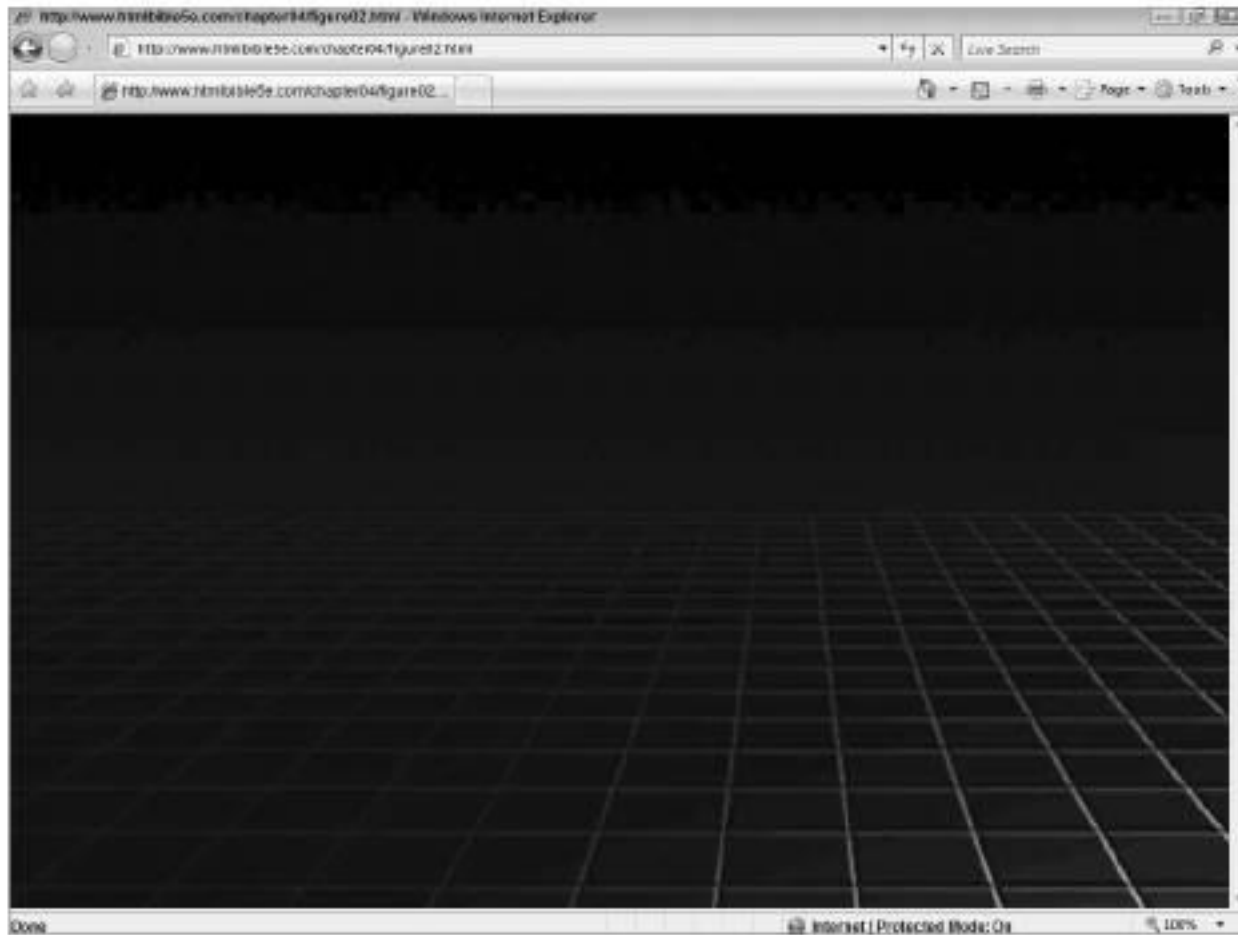
### Note

When you change the background color to a dark color or use a dark image, you should also change the text color so it will contrast with the background. For example, the following style sets the body background color to black, and the body text color to white:

```
<style>
  body { background-color: black; color: white;}
</style> ■
```

**FIGURE 4-2**

The grid in the document's background is courtesy of an image, grid.jpg.



## Summary

---

This chapter described the basic elements you need in all of your HTML documents. You learned how to set the document's title, how to include supplemental information in your documents using `<meta>` tags, how to use the `<base>` tag to set a default path for your document's URIs, and how to automatically refresh or redirect a document after a timed delay. The chapter also covered embedding scripts in HTML documents, placing style sections in a document, using profiles, and setting your document's background color and/or using an image as your background.

The next few chapters cover various formatting elements in more detail.

# Text Structuring Essentials

**T**he Web is used to transfer information in a variety of formats, but text is still the main form of communication across the Internet and even on the multimedia-rich World Wide Web. As such, it is important to understand the methods you can employ with HTML to format text.

This chapter covers the big picture — that is, text at the division and paragraph level. The next chapter delves into character formatting and other inline formatting concepts.

## Formatting Paragraphs

The most basic form to fit text within, whether in a book or on a Web page, is the paragraph. As you might have guessed, HTML supplies a specific tag to format text into discrete paragraphs.

The paragraph tags, `<p>` and `</p>`, provide the most basic block formatting for Web page text. Their use is straightforward: Place the opening tag (`<p>`) at the start of the paragraph and the ending tag (`</p>`) at the end of the paragraph. The user agent will format the paragraphs appropriately, usually by placing a blank line between them.

As an example, consider the following HTML code. Figure 5-1 shows the result of running this code in a browser.

```
<p>Welcome to The Oasis of Tranquility -- your source of day  
spa services at hair salon prices. Come visit us for that  
deep tissue or relaxing massage, facial, manicure, or  
hair coloring you have been putting off.</p>  
<p>Our concept is simple -- provide luxurious service affordable  
to most consumers. So stop in and let our experts please and  
pamper you today.</p>
```

### IN THIS CHAPTER

Formatting Paragraphs

Line Breaks

Divisions

Rules

Block Quotes

Preformatted Text

## Part I: Creating Content with HTML

---

**FIGURE 5-1**

Most user agents format paragraphs by placing a blank line between them.



You can, and typically should, use paragraph tags to encapsulate text even when the text is within a higher-level block tag. For example, note how the paragraph in the following code is placed within paragraph tags even though it is already within table data (<td>) tags:

```
<td>
  <p>Games with an "E" rating (Everyone) are suitable for
  consumers age 6 and older. These games may contain a
  minimal amount of mild violence (typically animated) and/or
  mild language.</p>
</td>
```

### Cross-Ref

The <td> and other table tags are discussed in detail in Chapter 9. ■

Other objects besides textual paragraphs can appear between paragraph tags also. For example, you might want an image to be vertically spaced evenly between two paragraphs. Placing the image tag within its own set of paragraph tags accomplishes that feat, as shown by the following code and corresponding rendering in the browser in Figure 5-2:

```
<p>Welcome to The Oasis of Tranquility -- your source of day
spa services at hair salon prices. Come visit us for that
deep tissue or relaxing massage, facial, manicure, or
hair coloring you have been putting off.</p>
```



```
<p></p>
<p>Our concept is simple -- provide luxurious service affordable
to most consumers. So stop in and let our experts please and
pamper you today.</p>
```

**FIGURE 5-2**

Paragraph tags aren't just for text; they can lend their block element encapsulation to any nonblock element.



## Line Breaks

Occasionally, you will want to break a line of text without ending the paragraph in which it appears. Reasons for doing so will vary, but typically the goal is to avoid the blank line that would result when the user agent displays your text.

For those times when you want to prematurely end a line but not the paragraph, use the line break tag (`<br />`).

An address block provides a good example of where to use the line break tag. Consider the following two address blocks and how they render in the browser shown in Figure 5-3:

```
<p>On Target Games</p>
<p>1 Target Place</p>
```



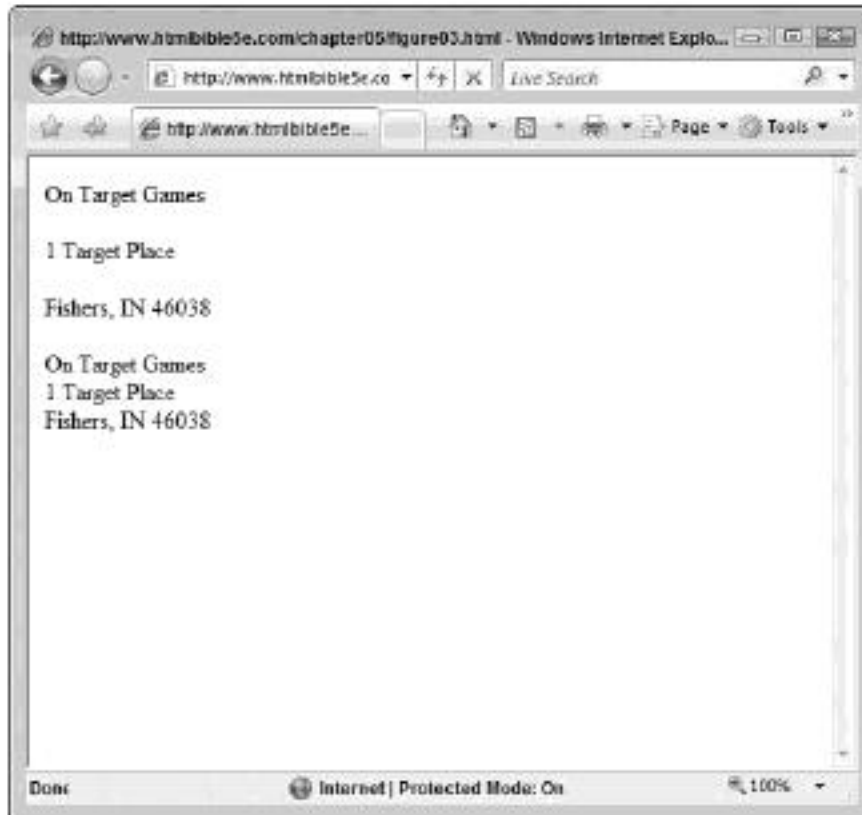
## Part I: Creating Content with HTML

---

```
<p>Fishers, IN 46038</p>
<p>On Target Games<br />
1 Target Place<br />
Fishers, IN 46038</p>
```

**FIGURE 5-3**

Using the line break tag enables you to keep your text tight while still being able to prematurely break lines.



Note that the line break tag does not have a closing mate and requires the ending slash. As such, the line break tag *does not* qualify as a block tag and must be contained within block tags. Notice in the preceding code example that the second address is still contained within paragraph tags — the line break tag is used within the paragraph to break lines.

## Divisions

---

Divisions are the big brother of paragraphs and are used to keep related objects (paragraphs, graphics, and so on) together. Divisions also allow the grouped objects to inherit most of the same formatting by applying the formatting to the division itself, which obviates the need to apply the formatting individually to each object contained within it.

### Cross-Ref

Applying styles to divisions and affecting the divisions' contents is covered in Chapter 34. ■

Division tags (<div> and </div>) are one of the highest-level block tags available in HTML. It is very typical to see HTML document bodies coded with blocks of divisions, similar to the following:

```
<body>
  <div>
    ...HTML content...
  </div>
  <div>
    ...some other HTML content...
  </div>
  <div>
    ...still some other HTML content...
  </div>
</body>
```

Because divisions are high-level block tags, they should be used to contain other block tags such as paragraph tags. While rarely done, placing a division within another block tag is not unheard of.

Conceptually, it helps to think of divisions as chapters in a book, keeping the paragraphs together. Better yet, given the rich visual nature of Web pages, think of divisions as defining the areas of a magazine page — the left-most column of text, the ad in the upper-right corner of the page, the right-most column of text, the author's bio block, and so on. In fact, given a few well-designed divisions and the appropriate content, you can design a Web page to resemble almost any magazine layout.

For example, consider the following example, which defines four divisions to encapsulate different content on the page. The result of this code is shown in Figure 5-4.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html>

<head>
<meta http-equiv="content-type"
  content="text/html; charset=iso-8859-1"/>
<meta name="description" content="description"/>
<meta name="keywords" content="keywords"/>
<meta name="author"
  content="Template design by Arcsin -
  http://templates.arcsin.se"/>
<link rel="stylesheet" type="text/css" href="default.css"
  media="screen"/>
<title>The Oasis of Tranquility</title>
</head>

<body>

<div class="container">
```

```
<div class="top">
   <!--
    <a href="index.html"><span>Bitter Sweet</span></a> -->
</div>

<div class="header"></div>

<div class="main">

  <div class="item">
    <div class="date">
      <div></div>
      <span></span>
    </div>
    <div class="content">
      <h1>About The Oasis of Tranquility</h1>
      <div class="body">
        <p>The Oasis of Tranquility is a premier spa, with an
        environment designed to embrace you in an air of calming
        relaxation. When you walk through our doors, the
        outside world will slip away, as you allow our
        dedicated team of professionals to pamper you in an
        experience focused on personalized care. Browse our site
        for the full experience that can be your own Oasis.</p>
      </div>
    </div>
  </div>

  <div class="item">
    <div class="date">
      <div>APR</div>
      <span>21</span>
    </div>
    <div class="content">
      <h1>2 For 1 Manicure Special</h1>
      <div class="body">
        <p>Treat a friend to the experience of the Oasis. Use the
        coupon below to receive two manicures for the price of one
        through the month of May.</p>
        <div class="coupon">
          <p>Present this coupon for a free manicure with the
          purchase of a similar manicure. Good during the same visit
          only, one per customer, other limitations may apply.</p>
          <p class="barcode">241 Manicure, valid May 2009 </p>
        </div>
      </div>
    </div>
  </div>
</div>
```

```
<div class="item">
  <div class="date">
    <div>APR</div>
    <span>10</span>
  </div>
  <div class="content">
    <h1>Visiting Hair Specialist</h1>
    <div class="body">
      <p>The Oasis is pleased to announce another visit from hair
        specialist Samual Hart.
        Make an appointment in the next two weeks for your
        personalized consultation on color, style, and product.</p>
    </div>
  </div>
</div>

</div> <!-- Main -->

<div class="navigation">

  <h1>Salon Services</h1>
  <ul>
    <li><a href="index.html">Hair Care</a></li>
    <li><a href="index.html">Nail Care</a></li>
  </ul>
  <h1>Spa Services</h1>
  <ul>
    <li><a href="index.html">Skin Care</a></li>
    <li><a href="index.html">Massage</a></li>
  </ul>
  <h1>Relaxation Amenities</h1>
  <ul>
    <li><a href="index.html">Amenities</a></li>
  </ul>
  <h1>About the Oasis</h1>
  <ul>
    <li><a href="index.html">About the Oasis</a></li>
    <li><a href="index.html">About the Staff</a></li>
  </ul>
</div>

<div class="clearer"><span></span></div>

<div class="footer">Copyright &copy; 2009 Oasis of Tranquility.
All Rights Reserved.</div>

</div> <!-- Container -->

</body>

</html>
```

## Part I: Creating Content with HTML

**FIGURE 5-4**

Many divisions can be created to hold various pieces of content on the same page.



## Rules

Rules are horizontal lines largely used to break sections of text into smaller chunks, or otherwise delimit the text in some way. For example, the headings in this book use rules under the heading text to set the heading off from the text to which it refers.

In HTML, rules are inserted into documents using the horizontal rule (`<hr />`) tag. Like many other empty tags that do not need to encapsulate anything, the `<hr />` tag is a solitary tag, having no closing mate, but therefore requiring the obligatory slash inside the tag.

An `<hr />` tag results in a gray, beveled line in most browsers, although the actual way the rule looks in the user agent is up to the agent itself. The rule results in a line break where the tag is placed, and the line itself stretches from margin to margin of the container (page, block, element, and so on) in which it is placed.

A default rule is shown in Figure 5-5.

**FIGURE 5-5**

A close-up of a horizontal rule rendered in Internet Explorer

The look of rules can be tailored to some extent using styles. For example, you can change the color using the `color` property, and the length of the line using the `width`. That being said, using styles to create and manipulate block element borders is generally a more flexible approach to custom rules.

### Cross-Ref

More information on the available style properties can be found in Part III of this book. ■

## Block Quotes

Occasionally, you will want to set blocks of text apart from the general text around it. One frequent example is the use of quotations, as shown in Figure 5-6.

**FIGURE 5-6**

Quotations or other text that needs to be set off from text around it can use blockquote tags



## Part I: Creating Content with HTML

---

The `blockquote` tag (`<blockquote>`) indents the elements it encapsulates and inserts space above and below the `blockquote` section, although the latter depends on the rules of the individual user agent rendering the element.

Consistency with the HTML standards and other documents is the main reason to continue to use the `<blockquote>` tag for quotations, rather than styled paragraphs, but you might find using styles a better, more flexible solution for your purposes.

### Note

Using the `<blockquote>` tag to simply indent text is discouraged in favor of using styles to accomplish such formatting. The use of `<blockquote>` should be limited to highlighting quoted text. ■

## Preformatted Text

---

User agents do a great job of optimizing text. Extra spaces are reduced to a single space, and redundant formatting is reduced or removed. However, you may sometimes want to preserve particular formatting in your text — keeping extra spacing, and so on.

The preformatted tags (`<pre>` and `</pre>`) can be used to encapsulate text for which you want the formatting preserved. Such text is generally space-formatted, columnar text, but is not limited to that type of text.

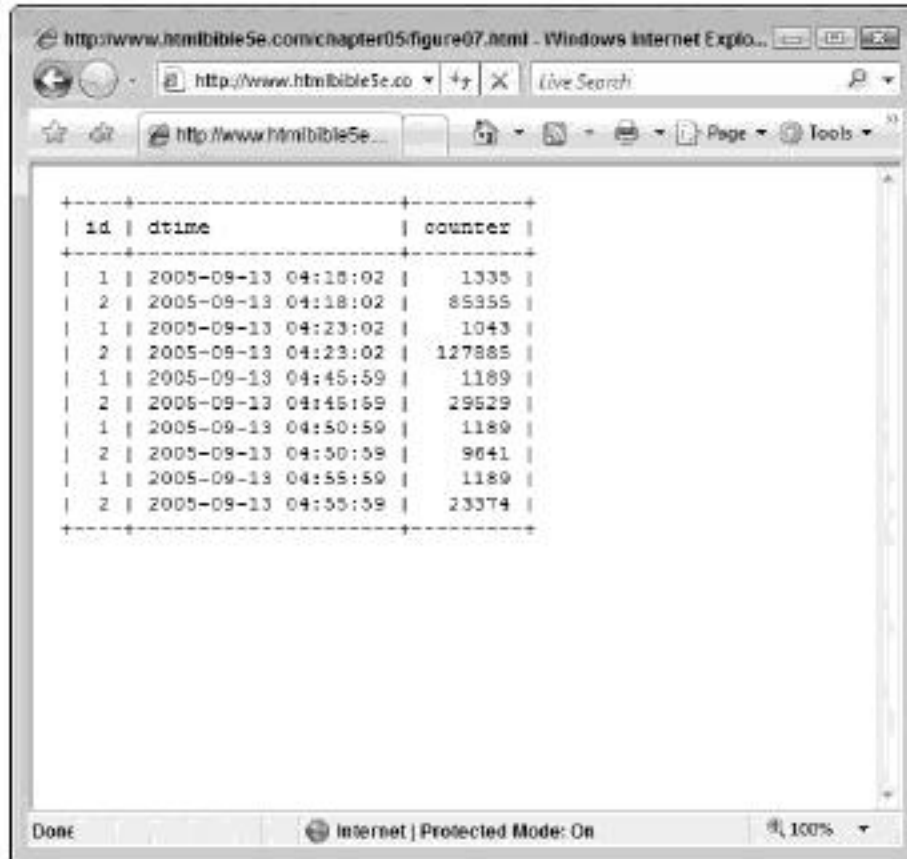
Note that the preformatted tag also causes the encapsulated text to be rendered in a monospace font to ensure that the character spacing does not change. The following code demonstrates an example of how preformatted tags can be used, and Figure 5-7 shows the results in a browser:

```
<pre>
+---+-----+-----+
| id | dttime           | counter |
+---+-----+-----+
| 1  | 2005-09-13 04:18:02 | 1335    |
| 2  | 2005-09-13 04:18:02 | 85355   |
| 1  | 2005-09-13 04:23:02 | 1043    |
| 2  | 2005-09-13 04:23:02 | 127885  |
| 1  | 2005-09-13 04:45:59 | 1189    |
| 2  | 2005-09-13 04:45:59 | 29529   |
| 1  | 2005-09-13 04:50:59 | 1189    |
| 2  | 2005-09-13 04:50:59 | 9641    |
| 1  | 2005-09-13 04:55:59 | 1189    |
| 2  | 2005-09-13 04:55:59 | 23374   |
+---+-----+-----+
</pre>
```



**FIGURE 5-7**

Preformatted tags enable you to include sections of text that the browser will not reformat.



## Summary

At this point in the book you should have a good understanding of how to construct an HTML standards-compliant document and format paragraphs of text within it. The next few chapters delve into specific character and special object formatting to give you fine control over your documents, make them look their best, and keep them standards compliant.

The next chapter dives into character formatting, showing you how to achieve a finer grain of formatting control over your text. This part then continues to cover other HTML formatting controls including lists (Chapter 7), links to other documents (Chapter 8), tables (Chapter 9), and frames (Chapter 10). By the end of this part, you should have a good understanding of the HTML elements at your disposal and the appropriate parts of a document on which to use them.



# Character Formatting Essentials

**A**lthough the modern-day Web is a haven of multimedia, text is still vitally important. Only through text can some messages be succinctly communicated. Even then, diversity in text can help further clarify a message. For example, emphasizing one word with bold or italic font can change the entire tone and meaning of a sentence.

This chapter discusses the tags you can use to format elements inside of block elements (characters, words, or sentences inside of paragraphs).

## Methods of Text Control

You can control the look and formatting of text in your documents using various means. It should come as no surprise that the more direct methods — `<font>` tags and the like — have been deprecated in favor of CSS controls in HTML 4.01 and XHTML. For historical context and completeness, the following sections cover the various means possible.

### Tip

Although it is sometimes easier to drop a direct formatting tag into text, resist the urge and use styles instead. Your documents will be more flexible and more standards-compliant. ■

## The `<font>` tag

The `<font>` tag enables you to directly affect the size and color of text. Intuitively, the `size` attribute is used to change the text's size; the `color` attribute is used to change the color. The size of the text is specified by a number, from 1 to 7, or by a signed number (also 1 to 7). In the latter

### IN THIS CHAPTER

Methods of Text Control

Bold and Italic Text

Use of Emphasis Instead of Italics

Monospace (Typewriter) Fonts

Superscripts and Subscripts

Abbreviations

Marking Editorial Insertions and Deletions

Grouping Inline Elements with the Span Tag

## Part I: Creating Content with HTML

---

case, the size change is relative to the size set by the `<basefont>` tag. The `<basefont>` tag has one attribute, `size`, which can be set to a number, 1 through 7.

### Note

Default font type and size is left up to the user agent. No standard correlation exists between the size used in a `<font>` tag and the actual font size used by the user agent. As such, the default size of the font (1 to 7) can vary considerably between user agents. ■

For example, if you wanted larger text in a red color, you could use a tag similar to the following:

```
<font size="+3" color="red">this is larger, red text</font>
```

Note that using `"+3"` for the size increases the text within the tag by a factor of 3 from the base font size.

The `<font>` tag is one of the HTML tags that have been deprecated in favor of styles. If you need to change the size of some of the text within a block element, use a `<span>` tag and styles instead. (The `<span>` tag is covered later in this chapter.)

## Emphasis and other text tags

You can use a handful of tags to emphasize portions of text. Although these tags have not been deprecated in HTML 4.01, it is strongly recommended that you make use of CSS instead, as CSS provides better control and flexibility, and the ability to cache formatting in style sheets for reuse.

Table 6-1 lists the emphasis tags and each one's use. A sample of their use is shown in Figure 6-1.

Because support for tags is somewhat haphazard, it is not standard across user agents — for example, you may not be able to tell the difference between text coded with `<cite>` or `<em>`.

## CSS text control

CSS provides several different properties to control text. Table 6-2 lists some of the more popular properties.

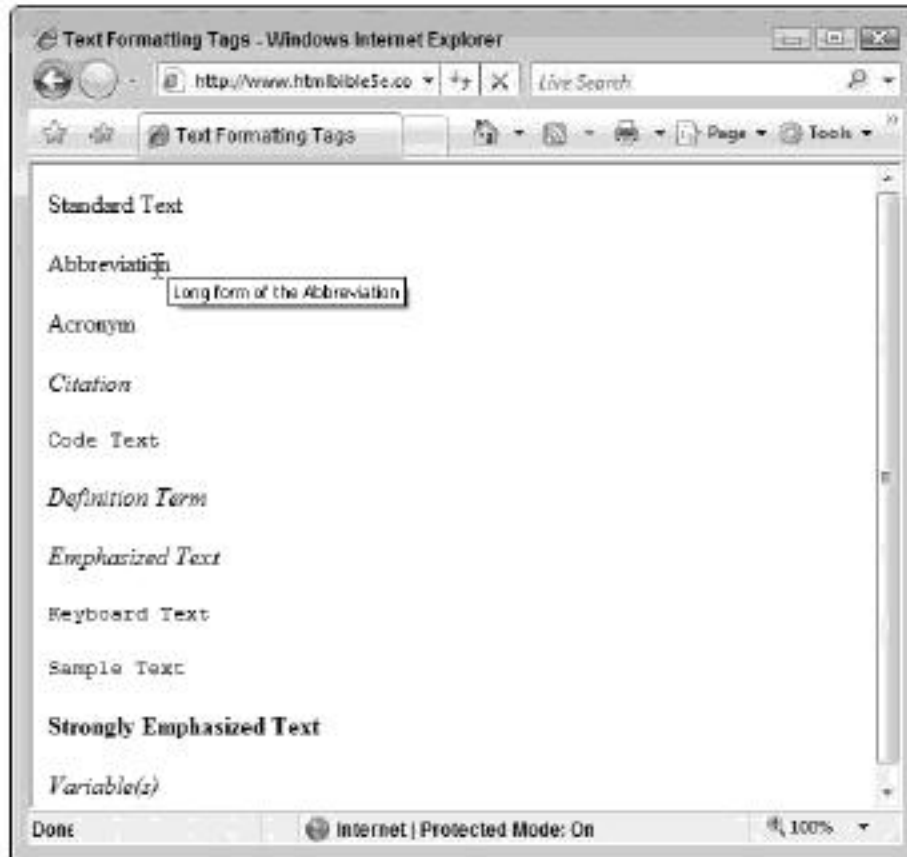
As you can see, CSS offers a bit more control over your text, enabling you to specify actual fonts and font sizes. However, the advantage to using CSS properties over hard-coded tags is not found in the list of available properties, but in the flexibility in formatting and affecting later changes — culled from the concept of keeping structure (HTML) and presentation (CSS) separate.

For example, suppose you were creating documentation for a programming language and wanted to format all reserved words a particular way — perhaps in a slightly larger, red, bold font. With tags, the code would resemble the following:

```
<p>The <font size="+1" color="red"><b>date</b></font>  
function can be used to ...
```

**FIGURE 6-1**

The different types of tags emphasizing text are rendered here.



Later, you might decide that the red color is too much emphasis, and larger, bold text is enough. You must then change every `<font>` tag used around reserved words, as follows, removing the color attribute:

```
<p>The <font size="+1"><b>date</b></font>  
function can be used to ...
```

Suppose, instead, that you used CSS, as shown in the following code:

```
<head>  
  <style type="text/css">  
    .reservedword { font: 14pt bold; color: red }  
  </style>  
</head>  
<body>  
<p>The <span class="reservedword">date</span> function can be  
used to...
```

If you later decided to change the formatting of reserved words, you would have to make only one change to the style definition at the top of the document. That one change would result in changing all instances of the style within the document.

**TABLE 6-1**

### Emphasis Tags

Tag	Use
<abbr>	Abbreviation
<acronym>	Acronym
<cite>	Citation
<code>	Code text
<dfn>	Definition term
<em>	Emphasized text
<kbd>	Keyboard text
<samp>	Sample text
<strong>	Strongly emphasized text
<var>	Variable(s)

**TABLE 6-2**

### CSS Text Properties

Property	Values	Use
color	color value	Change the color of text.
font	font-style font-variant font-weight font-size line-height font-family	Shortcut property for setting font style, variant, weight, size, line height, and font family.
font-family	family name value	Set the font family (face).
font-size	font size value	Set the font size.
font-style	normal   italic   oblique	Set font to italic.
font-variant	normal   small-caps	Set small caps.
font-weight	normal   bold   bolder   lighter	Set font to bold.
text-decoration	none   underline   overline   line-through   blink	Set under/overlining.
text-transform	none   capitalize   uppercase   lowercase	Transform font capitalization.

### Tip

If you used an external style sheet, that one change outlined in the preceding explanation could result in changing an unlimited number of documents that use the sheet. ■

## Bold and Italic Text

---

Two well-known text emphasis tags that survive in HTML are bold and italic. As used in the following code example, their effect on text is shown in Figure 6-2:

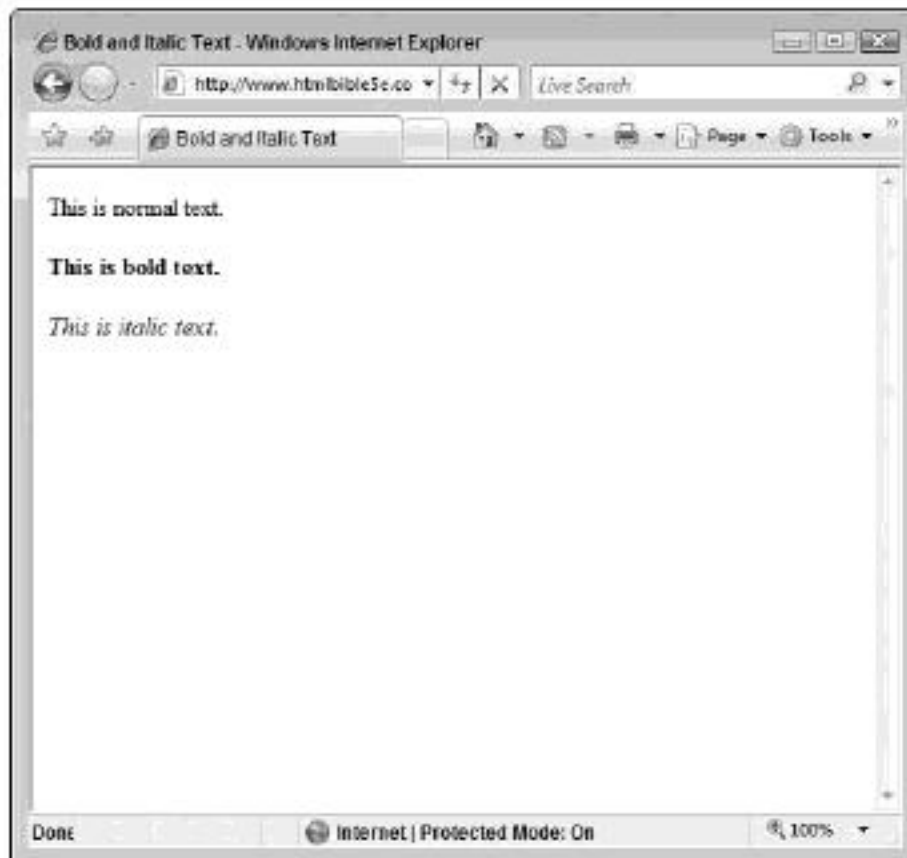
```
<p>This is normal text.</p>  
<p><b>This is bold text.</b></p>  
<p><i>This is italic text.</i></p>
```

### Note

Not every font has a bold and/or italic variant. When possible, the user agent will substitute a similar font when bold or italic is requested but not available. However, not all user agents are font-savvy. Your mileage with these tags may vary depending on the user agent being used, and the system on which it is being used. ■

**FIGURE 6-2**

Bold and italic tags at work





For the same reasons mentioned elsewhere, it is advisable to use CSS instead of hard-coded bold and italic tags.

## Use of Emphasis Instead of Italics

---

There is some common wisdom among Web developers that the emphasis tag (`<em>`) should be used instead of the italic tag (`<i>`). The rationale behind this opinion is that the italic tag should be used to *emphasize* text, not necessarily to italicize it (which has the notable side effect of emphasizing the very text it italicizes).

### Note

Although bold and italic tags have survived deprecation in HTML, their use is still discouraged in favor of CSS alternatives, for all the usual reasons highlighted throughout this chapter. ■

The one problem with specifying that text be italicized is that not all fonts have an italic variant. Some non-font-savvy user agents may choose to ignore the italic tag, rendering the text as normal, non-italicized, non-emphasized text. The emphasis tag, however, instructs the user agent to use its preferred method of emphasizing the coded text — italic, special symbols, a special font, bold characters, and so on. Regardless of the user agent, you can almost always rely upon text coded with the emphasis tag to appear differently than text not coded as such.

The bottom line: Use the emphasis tag when your goal is simply to have the coded text emphasized, regardless of how it is displayed by the user agent. Use the italic tag when you need the text to show up unconditionally as italic — if the user agent can render it as such.

## Monospace (Typewriter) Fonts

---

Another text formatting tag that has thus far survived deprecation is the teletype tag (`<tt>`). This tag is named for the teletype terminals used with the first computers, which were capable of printing only in a monospaced font. This tag tells the user agent that certain text should be rendered in a monospaced font. Suggested uses for this tag include reserved words in documentation, code listings, and so on. The following code shows an example of the teletype tag in use:

```
<p>Consider using the <tt>date</tt> function instead.</p>
```

### Tip

Again, the use of styles is preferred over individual inline tags. If you need text rendered in a monospace font, consider directly specifying the font parameters using styles instead of relying upon the `<tt>` tag. ■

# Superscripts and Subscripts

There are two tags, `<sup>` and `<sub>`, for formatting text in superscript and subscript, respectively. The following code shows an example of each tag, the output of which is shown in Figure 6-3:

```
<p>This is normal text.</p>  
<p>This is the 16<sup>th</sup> day of the month.</p>  
<p>Water tanks are clearly marked as H<sub>2</sub>O.</p>
```

**FIGURE 6-3**

Examples of superscript and subscript



# Abbreviations

You can use the abbreviation tag (`<abbr>`) to mark abbreviations, and, optionally, when using the `title` attribute, give readers the expansion of the abbreviation used. For example, you could use this tag with acronyms such as HTML:

```
<abbr title="Hypertext Markup Language">HTML</abbr>
```

Note that the expansion of the abbreviation is placed in the `<abbr>` tag's `title` attribute. Some user agents will display the value of the `title` attribute when the mouse/pointer is over the abbreviation, as previously shown in Figure 6-1. Other user agents may totally disregard the tag and its expansion `title` attribute.

The acronym tag (`<acronym>`) is very similar to the abbreviation tag but is used for acronyms. It, too, supports a `title` attribute for optionally supplying the expansion of the acronym.

## Marking Editorial Insertions and Deletions

---

To further strengthen the bond between HTML documents and printed material, the insert (`<ins>`) and delete (`<del>`) tags have been added to HTML. Both tags are used for redlining documents — that is, creating a visually marked-up document showing changes made to the document.

For example, the following code has been marked up with text to be inserted (underlined) and deleted (strikethrough). The output of this code is shown in Figure 6-4.

```
<p>Peter <del>are</del><ins>is</ins> correct. the proposal from Acme  
is lacking a few <del>minor </del>details.</p>
```

### Note

The underline tag (`<u>`) has been deprecated in favor of the insert tag (`<ins>`), and the strikethrough tag (`<strike>`) has been deprecated in favor of the delete tag (`<del>`). ■

## Grouping Inline Elements with the Span Tag

---

When using CSS for text formatting, you need a method to code text with the appropriate styles. If you are coding block elements, you can use the `<div>` tag to delimit the block, but with smaller chunks (inline elements) you should use `<span>`.

The `<span>` tag is used like any other inline tag (`<b>`, `<i>`, `<tt>`, and so on), surrounding the text/elements that it should affect. However, the `<span>` tag itself does not directly affect the text it encapsulates. You must use the `style` or `class` attribute to define what style(s) should be applied. For example, both of the following code paragraph samples would render the word **red** in a red-colored font:

```
<head>  
  <style type="text/css">  
    .redtext { color: red; }  
  </style>  
</head>
```

```
<body>
<!-- Paragraph 1, using direct style coding -->
<p>We should paint the document <span style="color: red">
red</span>.</p>
<!-- Paragraph 2, using a style class -->
<p>We should paint the document <span class="redtext">
red</span>.</p>
</body>
```

**FIGURE 6-4**

The ins and del tags can provide for suitable redlined documents.



Of the two methods, using the `class` attribute is preferred over using the `style` attribute because `class` avoids directly (and individually) coding the text. Instead, it references a separate style definition that can be repurposed with other text.

### Note

Throughout this chapter, I have advocated using styles in lieu of direct formatting using inline tags. The `<span>` tag is the vehicle you should use to accomplish that feat. For example, instead of coding individual instances of bold, italic text (`<b>``<em>`) throughout a document, create a style class using `font-weight` and `font-style` attributes and code each instance with a `<span>` tag that specifies that class. ■

### Summary

---

This chapter covered the formatting of text using inline tags. You learned two distinct methods (direct tags and styles) and the various tags to supplement textual formatting. Keep in mind that you should use `<div>` or other block tags to format larger sections of a document.

The following chapters (7 through 11) introduce you to larger formatting elements such as lists, tables, frames, and forms, and explain how to link documents to one another. All of this information gives you more formatting options for the text and character formatting techniques you have already learned.

# Lists

**H**TML and its various derivatives were originally intended to reproduce academic and research text. For this reason, particular care was taken to ensure that specific elements, such as lists and tables, were implemented and robust enough to handle the tasks for which they serve.

In the case of lists, HTML defines three different types of lists: *ordered lists* (numbered), *unordered lists* (bulleted), and *definition lists* (term and definition pairs). This chapter covers all three types of lists and the various syntax and formatting possibilities of each.

## IN THIS CHAPTER

Understanding Lists

Ordered (Numbered) Lists

Unordered (Bulleted) Lists

Definition Lists

Nested Lists

## Understanding Lists

All lists, whether ordered, unordered, or definition, share similar elements. Each HTML list has the following structure:

```
<list_tag>
  <item_tag>Item text</item_tag>
  <item_tag>Item text</item_tag>
  ...
</list_tag>
```

### Note

Definition lists are slightly different in syntax because they use a term tag (<dt>) and a definition description tag (<dd>). See the “Definition Lists” section later in this chapter for more information. ■

For each list, you need the list opening tag, a corresponding closing tag, and individual item tags for each element actually in the list. Essentially,

## Part I: Creating Content with HTML

---

the entire list must be delimited by list open and close tags, with list items appearing between the two tags with open and close tags of their own. This structure will become abundantly clear throughout the chapter.

Each type of list has its own display format:

- An ordered list precedes its items with a number or letter.
- An unordered list precedes its items with a bullet (as in this list).
- A definition list has two pieces for each item: a term and a definition.

The ordered and unordered lists have many different display options available:

- Ordered lists can have their items preceded by the following:
  - Arabic numbers
  - Roman numerals (uppercase or lowercase)
  - Letters (uppercase or lowercase)
  - Numerous other language-specific numbers/letters
- Unordered lists can have their items preceded by the following:
  - Several styles of bullets (filled circle, open circle, square, and so on)
  - Images

More information on the individual list types is provided in the following sections.

## Ordered (Numbered) Lists

---

Ordered lists have elements that are preceded by numbers or letters. They are meant to provide a sequence of ordered steps for an activity. For example, this book uses numbered lists when stepping you through a process. Such a list might resemble the following:

1. Press and hold the reset button until the power light blinks rapidly.
2. Release the reset button.
3. Wait until the power light returns to a steady state.

Ordered lists use the ordered list tag (`<ol>`) to delimit the entire list, and the list item tag (`<li>`) to delimit each individual list item.

In the preceding example, the list has three elements numbered with Arabic numbers. This is the default for ordered lists in HTML, as shown in the following code, whose output is shown in Figure 7-1:

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