Chapter 5

Text and Lists

In This Chapter

- Working with basic blocks of text
- Manipulating text blocks
- Creating bulleted, numbered, and definition lists

HTML documents consist of text, images, multimedia files, links, and other pieces of content that you bring together into one page by using markup elements and attributes. You use blocks of text to create such document elements as headings, paragraphs, and lists. The first step in creating a solid HTML document is laying a firm foundation that establishes the document's structure.

Formatting Text

Here's a super-ultra-technical definition of a *block of text:* some chunk of content that wraps from one line to another inside an HTML element.

Your HTML page is a giant collection of blocks of text:

- Every bit of content on your Web page must be part of some block element.
- Every block element sits within the `<body>` element on your page.

HTML recognizes several kinds of text blocks that you can use in your document, including (but not limited to)

- Paragraphs
- Headings
- Block quotes
- Lists
- Tables
- Forms

Inline elements versus text blocks

The difference between inline elements and a block of text is important. HTML elements in this chapter describe blocks of text. An *inline element* is a word or string of words *inside* a block element (for example, text emphasis elements such as or). Inline elements must be nested within a block element; otherwise, your HTML document isn't syntactically correct.

Inline elements, such as linking and formatting elements, are designed to link from or change the appearance of a few words or lines of content found inside those blocks.

Paragraphs

Paragraphs are used more often in Web pages than any other kind of text block.

HTML browsers don't recognize the hard returns that you enter when you create your page inside an editor. You must use a element to tell the browser to separate the contained block of text as a paragraph.

Formatting

To create a paragraph, follow these steps:

1. **Add in the body of the document.**
2. **Type the content of the paragraph.**
3. **Add to close that paragraph.**

Here's what it looks like:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>All About Blocks</title>
  </head>

  <body>
    <p>This is a paragraph. It's a very simple structure that you will use
       time and again in your Web pages.</p>
    <p>This is another paragraph. What could be simpler to create?</p>
  </body>
</html>
```

This HTML page includes two paragraphs, each marked with a separate element. Most Web browsers add a line break and full line of white space after every paragraph on your page, as shown in Figure 5-1.

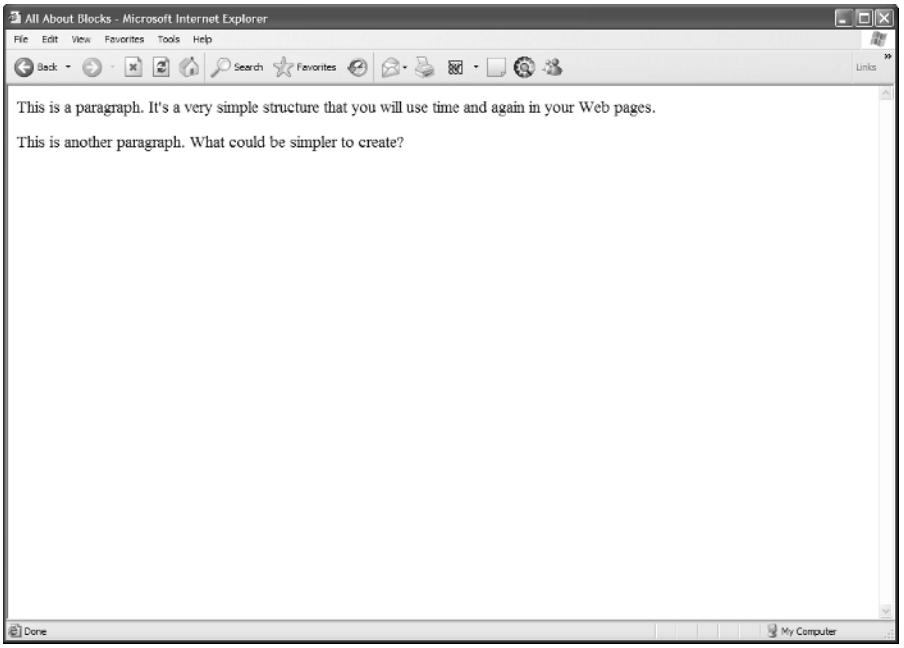

Figure 5-1: Web browsers delineate paragraphs with line breaks.

Some people don't use the closing `</p>` tag when they create paragraphs. Although some browsers let you get away with this, leaving out the closing tag

- Doesn't follow correct syntax
- Causes problems with style sheets
- Can cause a page to appear inconsistently from browser to browser

You can control the formatting (color, style, size, and alignment) of your paragraph by using Cascading Style Sheets (CSS), which we cover in Chapters 8 and 9.

Alignment

By default, the paragraph aligns to the left. You can use the `align` attribute with a value of `center`, `right`, or `justify` to override that default and control the alignment for any paragraph.

```
<p align="center">This paragraph is centered.</p>
<p align="right">This paragraph is right-justified.</p>
<p align="justify">This paragraph is double-justified.</p>
```

Figure 5-2 shows how a Web browser aligns each paragraph according to the value of the `align` attribute.

The `align` attribute has been deprecated (rendered obsolete) in favor of using CSS (see Chapter 8).

Headings

Headings break a document into sections. This book uses headings and subheadings to divide every chapter into sections, and you can do the same with your Web page. Headings can

- Create an organizational structure
- Break up the visual appearance of the page
- Give visual clues about how the pieces of content are grouped

HTML includes six elements to help you define six different heading levels in your documents:

- `<h1>` is the most prominent heading (Heading 1)
- `<h6>` is the least prominent heading (Heading 6)

Follow heading order from highest to lowest as you use HTML heading levels. That is, don't use a second-level heading until you've used a first-level heading, don't use a third-level heading until you've used a second, and so on. If you want to change how headings appear in a browser, Chapter 8 and Chapter 9 show you how to use style sheets.

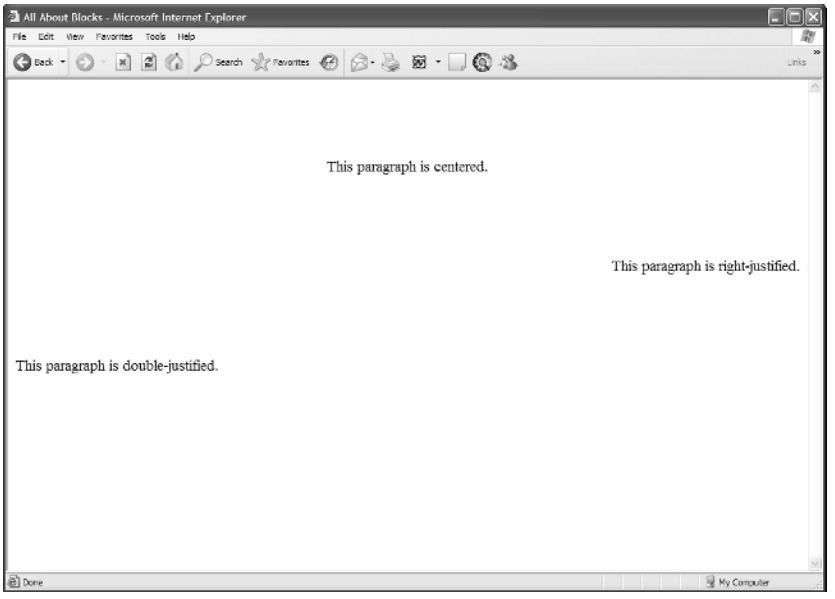

Figure 5-2: Use the align attribute with a paragraph to specify the horizontal alignment.

Formatting

To create a heading, follow these steps:

1. **Add <h*n*> in the body of your document.**
2. **Type the content for the heading.**
3. **Add </h*n*>.**

Browser displays

Every browser has a different way of displaying heading levels, and we cover that in the following two sections.

Graphical browsers

Most graphical browsers use a distinctive size and typeface for headings:

- First-level headings (<h1>) are the largest (usually two or three font sizes larger than the default text size for paragraphs).
- Sixth-level headings (<h6>) are the smallest and may be two or three font sizes *smaller* than the default paragraph text.

The following excerpt of HTML markup shows all six headings at work:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>All About Blocks</title>
  </head>

  <body>
    <h1>First-level heading</h1>
    <h2>Second-level heading</h2>
    <h3>Third-level heading</h3>
    <h4>Fourth-level heading</h4>
    <h5>Fifth-level heading</h5>
    <h6>Sixth-level heading</h6>
  </body>
</html>
```

Figure 5-3 shows this HTML page as rendered in a browser.

You can use CSS to format such heading aspects as color, size, line height, and alignment.

By default, most browsers use Times New Roman fonts for all headings. The font size decreases as heading level increases. (Default sizes for first- through sixth-level headings are, respectively, 24, 18, 14, 12, 10, and 8.) You can override any of this formatting by using CSS.

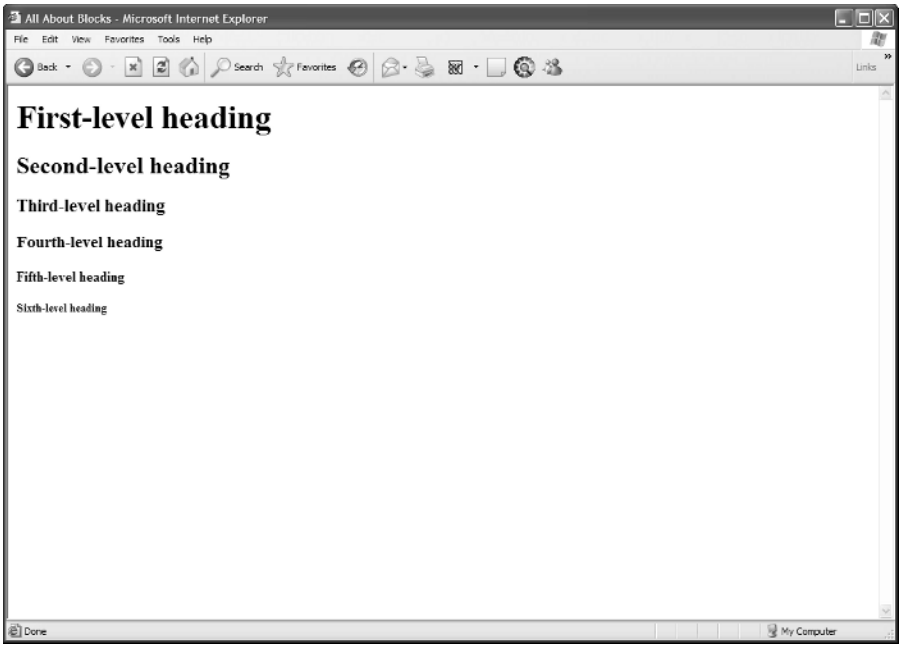

Figure 5-3: Web browsers display headings in decreasing size from level one to level six.

Text browsers

Text-only browsers use different heading conventions than graphical browsers because text-only browsers display all content using a single size and font.

Controlling Text Blocks

Blocks of text are the foundation for your page. You can break those blocks to better guide readers through your content.

Block quotes

A *block quote* is a long quotation or excerpt from a printed source that you set apart on your page. You use the `<blockquote>` element to identify block quotes:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
       "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Famous Quotations</title>
  </head>
```

```
  <body>
    <h1>An Inspiring Quote</h1>
    <p>When I need a little inspiration to remind me of why I spend my days
       in the classroom, I just remember what Lee Iococca said:</p>
    <blockquote>
      In a completely rational society, the best of us would be teachers
      and the rest of us would have to settle for something else.
    </blockquote>
  </body>
</html>
```

Most Web browsers display block-quote content with a slight left indent, as shown in Figure 5-4.

Preformatted text

Ordinarily, HTML ignores white space inside documents. A browser won't display a block element's

- Hard returns
- Line breaks
- Large white spaces

The following markup includes several hard returns, line breaks, and a lot of space characters. Figure 5-5 shows that the Web browser ignores all of this.

```
<p>This is a paragraph

   with a lot of white space

         thrown in for fun (and as a test of course).</p>
```

The preformatted text element (<pre>) instructs browsers to keep all white space intact as it displays your content (like the following sample). Use the <pre> element in place of the <p> element to make the browser apply all your white space, as shown in Figure 5-6.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>White space</title>
  </head>

  <body>
    <pre>This is a paragraph
```

```
      with a lot of white space

      thrown in for fun (and as a test of course).
    </pre>
  </body>
</html>
```

You may want the browser to display white spaces in an HTML page where proper spacing is important, such as

- Code samples
- Text tables

You can nest `<pre>` elements inside `<blockquote>` elements to carefully control how the lines of quoted text appear on the page.

Line breaks

By default, browsers usually *wrap* text that appears in block elements, such as paragraphs, headings, and block quotes. If a text line reaches the end of a browser window, the next word automatically starts a new line. You can manually control the end of a text line with a *line break* (denoted by the `<br />` element).

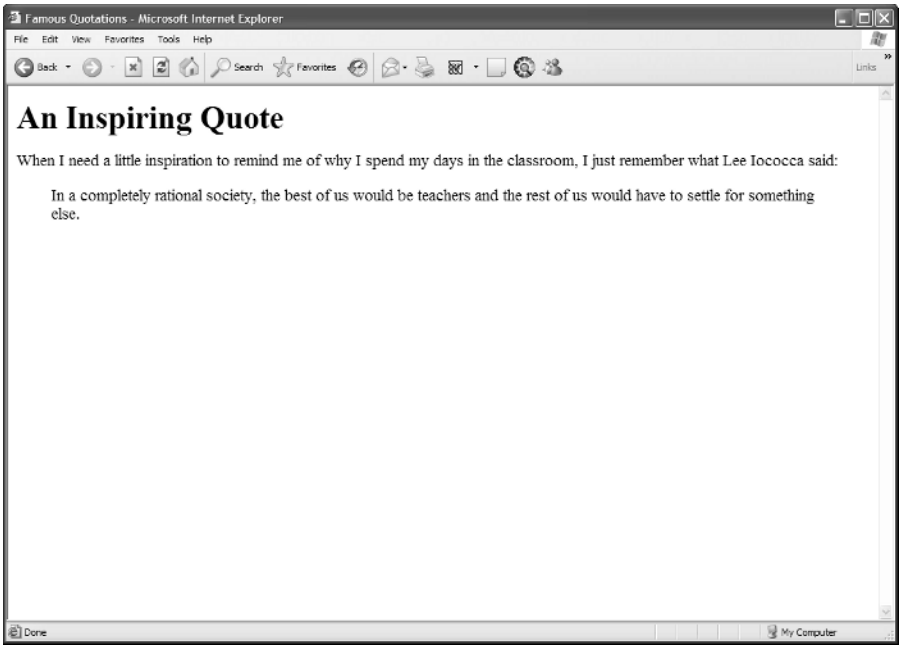

Figure 5-4: Web browsers typically indent a block quote to separate it from paragraphs.

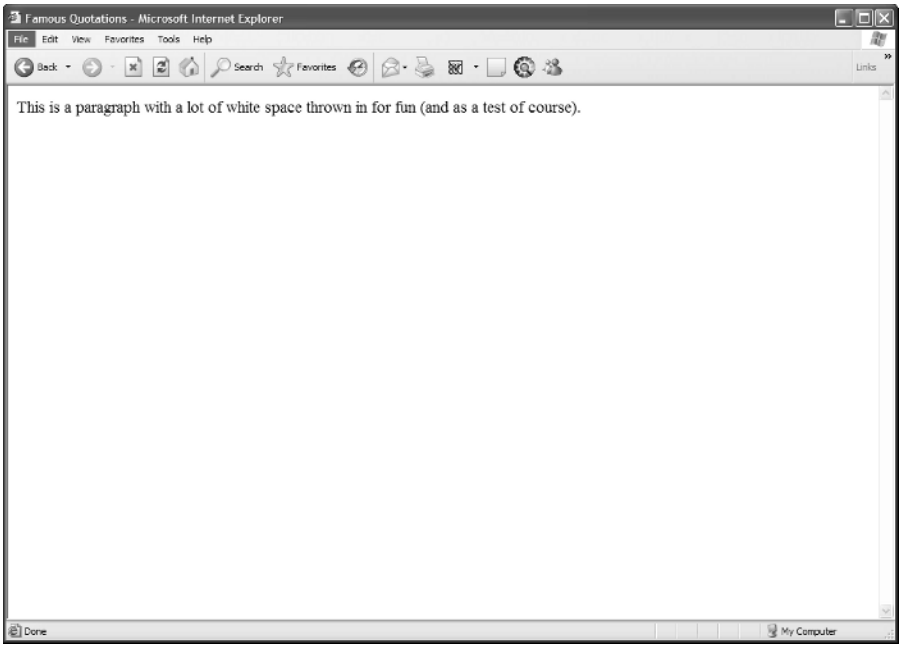

Figure 5-5: Web browsers routinely ignore white space.

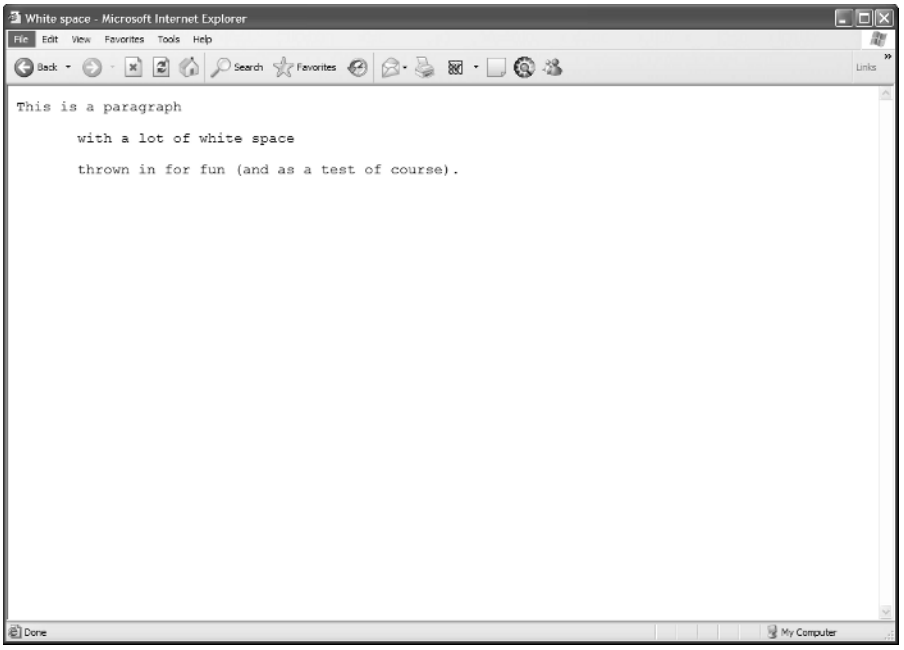

Figure 5-6: Use preformatted text to force browsers to recognize white space.

Function

The
 element is the HTML equivalent of the manual line break that you use in paragraphs and other blocks of text when you're working in a word-processing program. When a browser sees a
, it ends the line there and starts the next line.

The difference between a line break and a paragraph is that a line break doesn't use any special formatting that you can apply at the end or beginning of a paragraph, such as

- Extra vertical space
- First-line indenting

Formatting

The following markup formats the lines of text in a poem with line breaks. The entire poem is described *as a single paragraph,* and the
 element marks the end of each line:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title> Shakespeare in HTML</title>
  </head>

  <body>
  <h1>Shakespeare's Sonnets XVIII: Shall I compare thee to a summer's day? </h1>
    <p>
      Shall I compare thee to a summer's day? <br />
      Thou art more lovely and more temperate. <br />
      Rough winds do shake the darling buds of May, <br />
      And summer's lease hath all too short a date. <br />
      Sometime too hot the eye of heaven shines, <br />
      And often is his gold complexion dimm'd; <br />
      And every fair from fair sometime declines, <br />
      By chance or nature's changing course untrimm'd; <br />
      But thy eternal summer shall not fade <br />
      Nor lose possession of that fair thou ow'st; <br />
      Nor shall Death brag thou wander'st in his shade, <br />
      When in eternal lines to time thou grow'st: <br />
      So long as men can breathe or eyes can see, <br />
      So long lives this, and this gives life to thee. <br />
    </p>
  </body>
</html>
```

Figure 5-7 shows how a browser handles each line break. In this example, the poem isn't left-indented because the <p> element replaces the <blockquote> element.

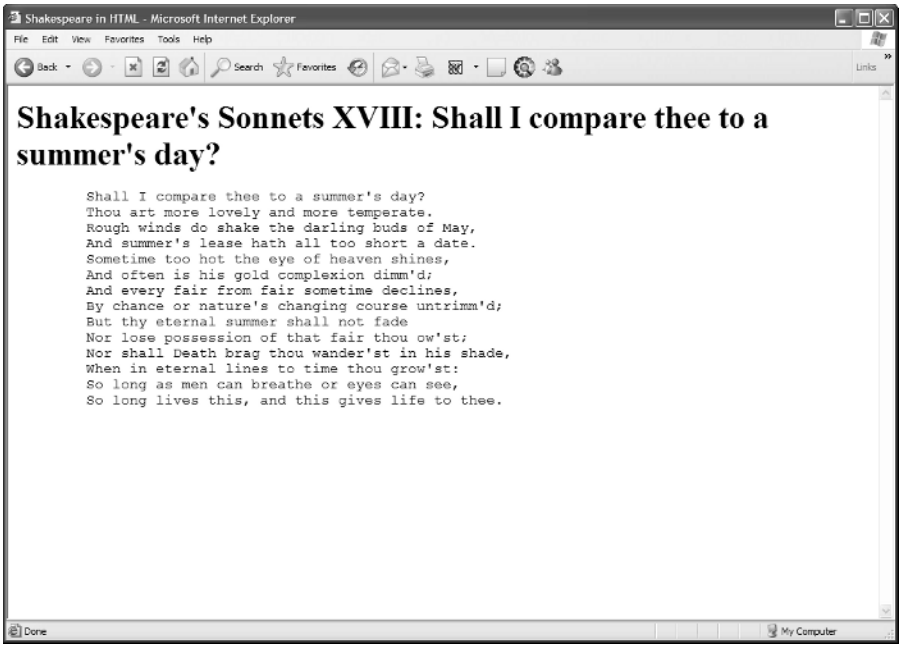

Figure 5-7: Using the
 element to specify where lines in block elements should break.

Horizontal rules

The horizontal rule element (`<hr />`) helps you include solid straight lines *(rules)* on your page.

The browser creates the rule based on the `<hr />` element, so users don't wait for a graphic to download. A horizontal rule is a good option to

- Break your page into logical sections.
- Separate your headers and footers from the rest of the page.

Formatting

When you include an `<hr />` element on your page, like the following HTML, the browser replaces it with a line, as shown in Figure 5-8.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Horizontal Rules</title>
  </head>
```

```
  <body>
    <p>This is a paragraph followed by a horizontal rule.</p>

    <hr />

    </p>This is a paragraph preceded by a horizontal rule.</p>
  </body>
</html>
```

A horizontal rule must always sit on a line by itself; you can't add the `<hr />` element in the middle of a paragraph (or other block element) and expect the rule to just appear in the middle of the block.

Attributes

Four different attributes control the appearance of each horizontal rule:

- `width`: Specifies line width either in *pixels* or by *percentage of display area width* (which we call "the page" in discussion that follows).

  For example, a rule can be 50 pixels wide or take 75 percent of the page.
- `size`: Specifies the height of the line in pixels. The default is 1 pixel.
- `align`: Specifies the horizontal alignment of the rule as either `left` (the default), `center`, or `right`.

  If you don't define a width for your rule, it takes the entire width of the page. The alignment won't make any difference.
- `noshade`: Specifies a solid line with no shading.

  By default, most browsers display hard rules with a shade.

These formatting attributes are deprecated in favor of using CSS.

This bit of HTML creates a horizontal rule that takes up 45 percent of the page, is 4 pixels high, aligned to the center, and has shading turned off:

```
    <p>This is a paragraph followed by a horizontal rule.</p>

    <hr width="45%" size="4" align="center" noshade="noshade" />

    <p>This is a paragraph preceded by a horizontal rule.</p>
```

Figure 5-9 shows how the addition of these attributes can alter how a browser displays the rule.

Figure 5-10 shows how you can use horizontal rules in the real world to highlight important content. The LANWrights, Inc., Web site uses colored hard rules to frame a key statement on the site's home page. The rules make the statement stand out from the rest of the page.

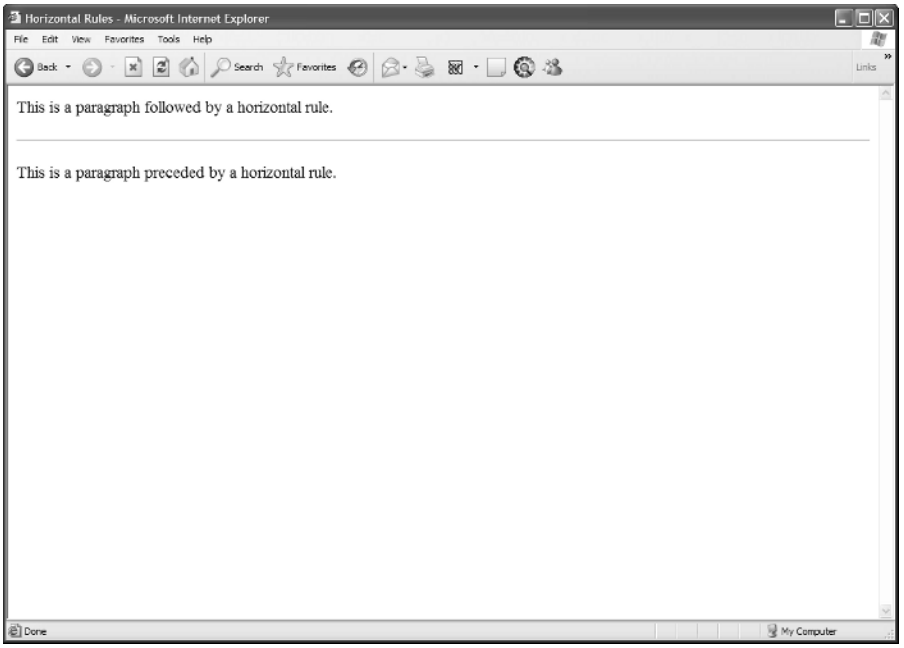

Figure 5-8: Use the <hr /> element to add horizontal lines to your page.

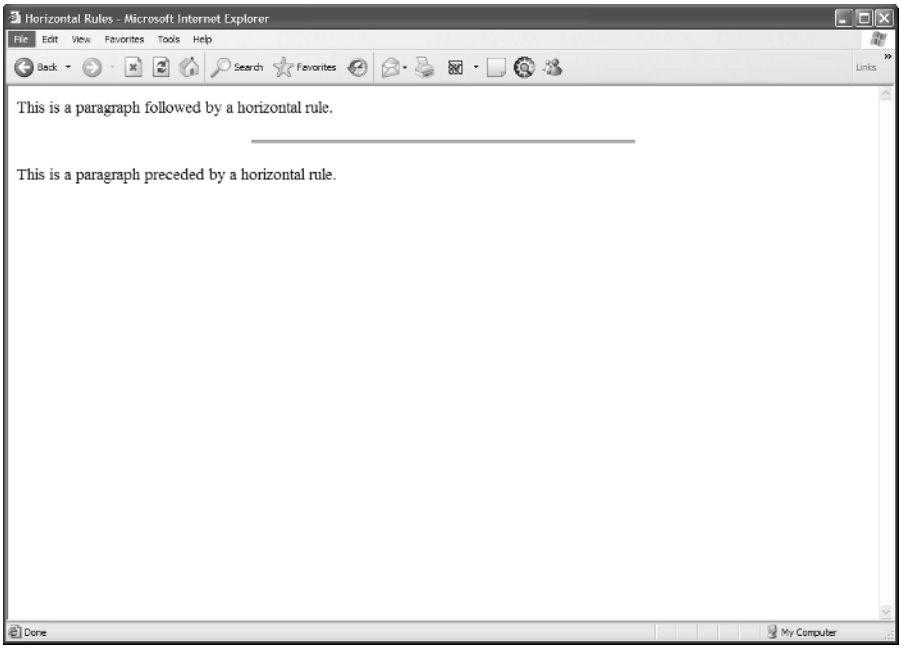

Figure 5-9: Use the <hr /> attributes to better control how a browser displays the rule.

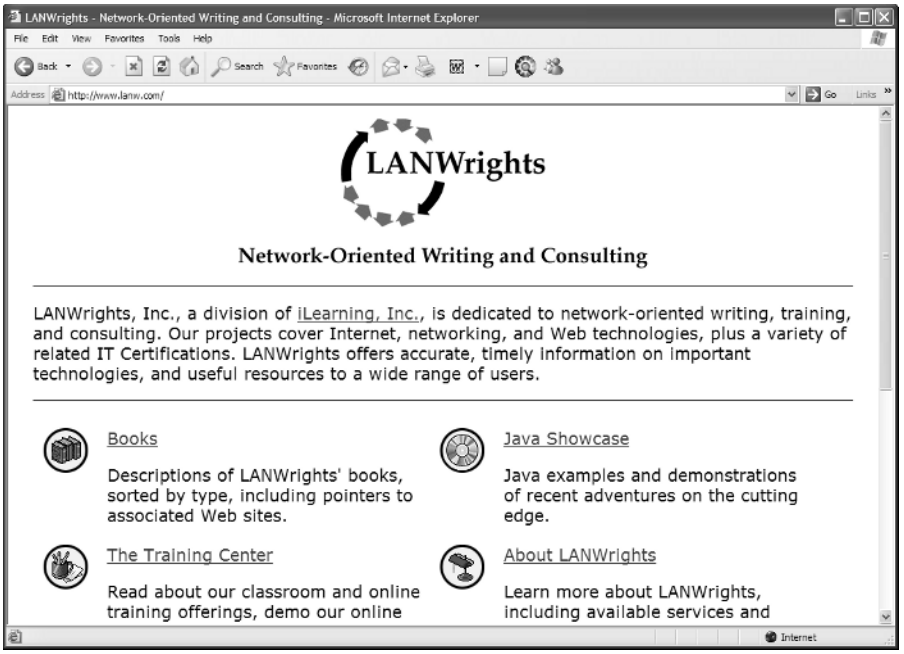

Figure 5-10: The LANWrights, Inc., Web site uses hard rules to draw your attention to important information on the page.

CSS gives you much more control over the placement of horizontal rules; you can even fancy them up with color and shading options.

Organizing Information

Lists are powerful tools for arranging similar elements together, and they give visitors to your site an easy way to hone in on groups of information. You can put just about anything in a list, from a set of instructions to a collection of navigational hyperlinks.

Lists use a combination of elements — at least two components:

- A markup element that says "Hey browser! The following items are a list."
- Markup elements that say "Hey browser! This is an item in the list."

HTML provides for three different kinds of lists:

- Numbered lists
- Bulleted lists
- Definition lists

Numbered lists

A *numbered list* consists of at least two items, each prefaced by a number. Usually, a person numbers a list when the order of the items is important.

You use two kinds of elements for a numbered list:

- The ordered list element (`<ol>`) specifies that this is a numbered list.
- List item elements (`<li>`) mark each item in the list.

Formatting

A numbered list with three items requires elements and content in the following order:

1. `<ol>`
2. `<li>`
3. Content for the first list item
4. `</li>`
5. `<li>`
6. Content for the second list item
7. `</li>`
8. `<li>`
9. Content for the third list item
10. `</li>`
11. `</ol>`

The following markup defines a three-item numbered list:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Numbered Lists</title>
  </head>

  <body>
    <h1>Things to do today</h1>
    <ol>
      <li>Feed cat</li>
```

```
      <li>Wash car</li>
      <li>Grocery shopping</li>
    </ol>
  </body>
</html>
```

Figure 5-11 shows how a browser renders this markup. You don't actually have to specify a number for each item in the list; the browser identifies the list items from the markup and adds the numbers.

If you swap the first two items in the list, they're still numbered in order when the page appears, as shown in Figure 5-12.

```
    <ol>
      <li>Wash car</li>
      <li>Feed cat</li>
      <li>Grocery shopping</li>
    </ol>
```

Numbering

Two different `<ol>` element attributes control the appearance of a numbered list:

- `start`: Specifies the first number in the list.
  - The default starting number is 1.

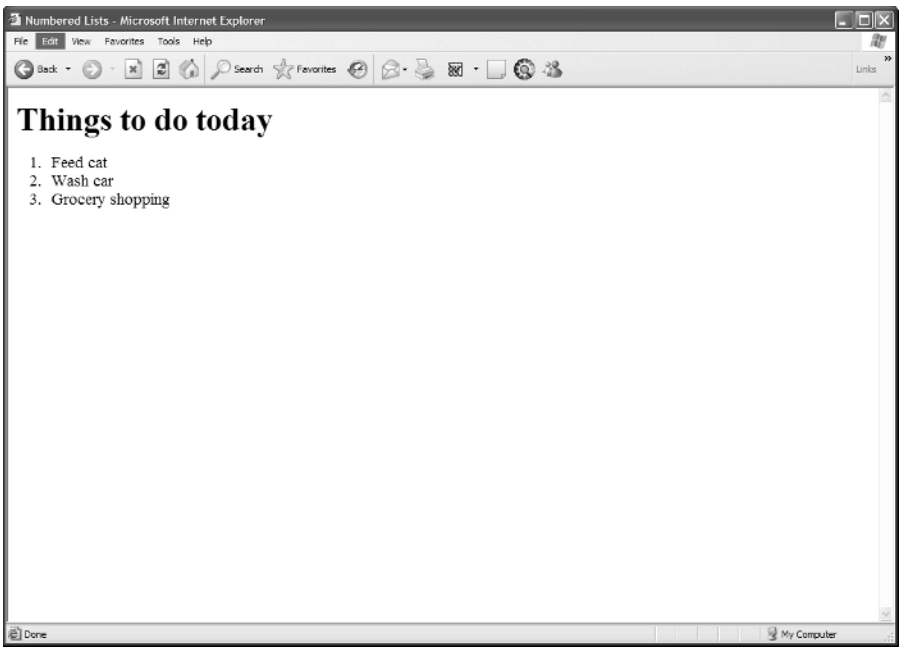

Figure 5-11: Use the <ol> and <li> tags to create a numbered list.

- You can specify any number as the start number for the new list.

  Specify a start number when you resume a list after an unnumbered paragraph or other block element.

- `type`: Specifies the numbering style from the list. You can choose from five predefined numbering styles:
  - `1`: Decimal numbers.
  - `a`: Lowercase letters.
  - `A`: Uppercase letters.
  - `i`: Lowercase Roman numerals.
  - `I`: Uppercase Roman numerals.

The following markup uses ordered list elements and attributes to create a list that uses uppercase Roman numerals and begins numbering at 5 (V in Roman numerals):

```
<ol start="5" type="I">
  <li>Wash car</li>
  <li>Feed cat</li>
  <li>Grocery shopping</li>
</ol>
```

Figure 5-13 shows how the attributes affect the list's appearance in a browser.

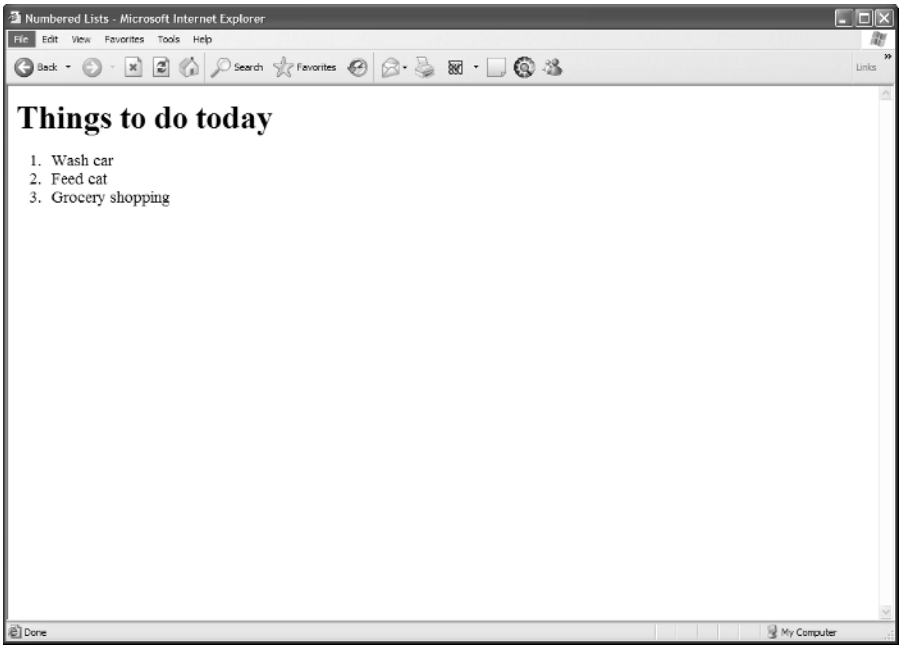

Figure 5-12: Web browsers set the numbers for your list according to the order items appear in the list.

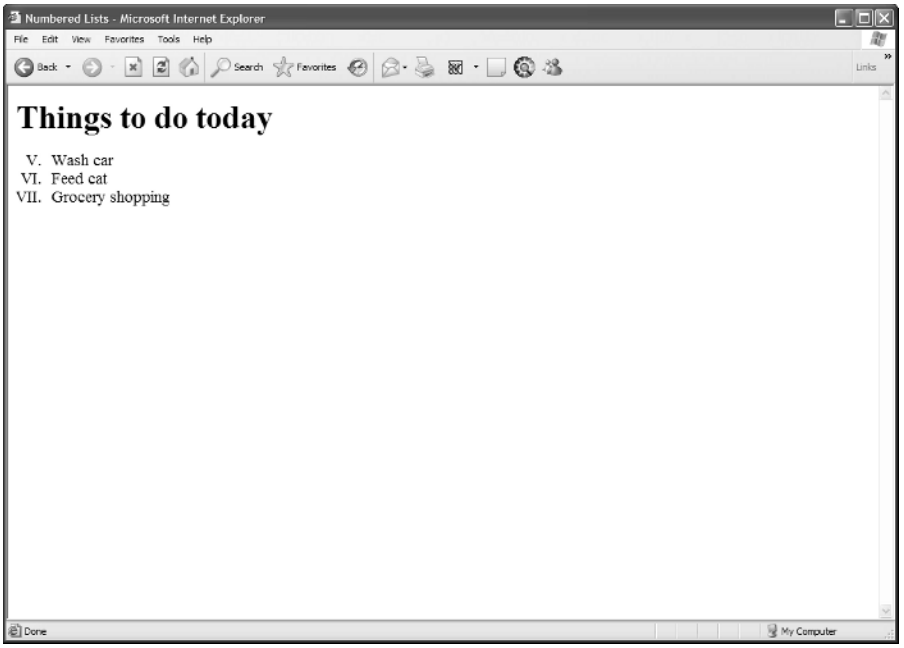

Figure 5-13: The start and type attributes guide the appearance of a numbered list in a browser.

You have more control over your lists if you use CSS to define formatting. That's why the `start` and `type` attributes for list markup are deprecated; see Appendix A for information about deprecated attributes.

Bulleted lists

A *bulleted list* consists of one or more items each prefaced by a *bullet* (often a big dot; this book uses *check marks* as bullets).

You use this type of list if the order of the presentation of the items isn't necessary for understanding the information presented.

Formatting

A bulleted list requires the following:

- The unordered list element (`<ul>`) specifies that you're creating a bulleted list.
- A list item element (`<li>`) marks each item in the list.
- The closing tag for the unordered list element (`</ul>`) indicates that the list has come to its end.

An unordered list with three items requires elements and content in the following order:

1. `<ul>`
2. `<li>`
3. Content for the first list item
4. `</li>`
5. `<li>`
6. Content for the second list item
7. `</li>`
8. `<li>`
9. Content for the third list item
10. `</li>`
11. `</ul>`

The following markup formats a three-item list as a bulleted list:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Bulleted Lists</title>
  </head>

  <body>
    <h1>Things to do today</h1>
    <ul>
      <li>Feed cat</li>
      <li>Wash car</li>
      <li>Grocery shopping</li>
    </ul>
  </body>
</html>
```

Figure 5-14 shows how a browser renders this with bullets.

Styles

You can use the `type` attribute (deprecated) with the `<ul>` element to specify what kind of bullet you want the list to use.

- `disc`: Solid circle bullets (the default)
- `square`: Solid square bullets
- `circle`: Hollow circle bullets

Figure 5-14: An unordered list uses bullets instead of numbers to mark items.

The addition of the `type` attribute to the bulleted-list markup just given changes the bullets from discs to squares, as shown in Figure 5-15. Here's what the relevant markup looks like:

```
<ul type="square">
  <li>Feed cat</li>
  <li>Wash car</li>
  <li>Grocery shopping</li>
</ul>
```

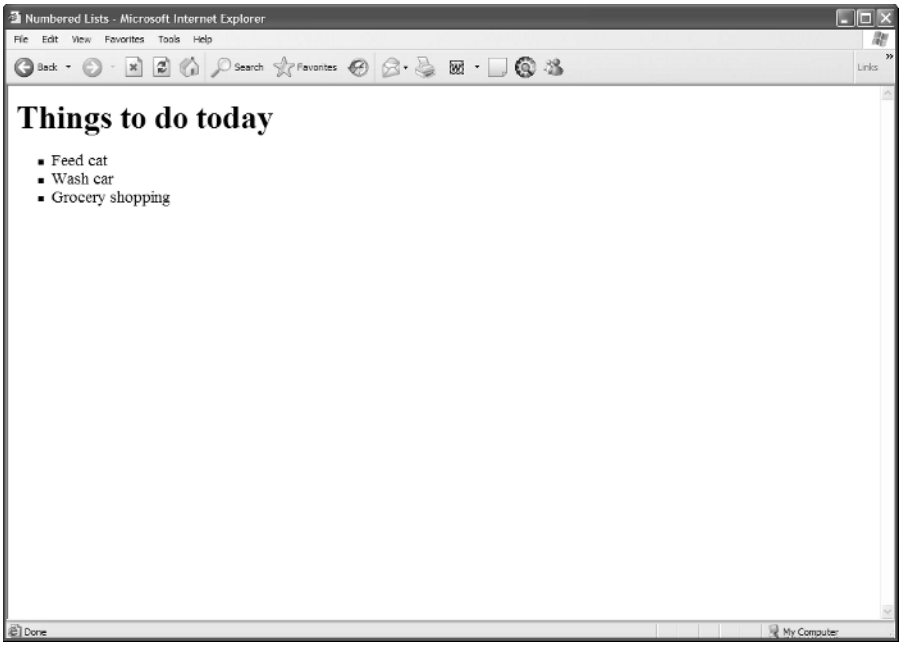

Figure 5-15: Use the type attribute to change the bullet style for an unordered list.

Use CSS if you want more control over the formatting of your lists.

Definition lists

Definition lists group terms and definitions into a single list and require three different elements to complete the list:

- `<dl>`: Holds the list definitions.
- `<dt>`: Defines a term in the list.
- `<dd>`: Defines a definition for a term.

You can have as many terms (defined by `<dt>`) in a list as you need. Each term can have one or more definitions (defined by `<dd>`).

To create a definition list with two items requires elements and content in the following order:

1. `<dl>`
2. `<dt>`
3. First term name
4. `</dt>`
5. `<dd>`
6. Content for the definition of the first item
7. `</dd>`
8. `<dt>`
9. Second term name
10. `</dt>`
11. `<dd>`
12. Content for the definition of the second item
13. `</dd>`
14. `</dl>`

The following definition list includes three terms, one of which has two definitions:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
```

```
    <title>Definition Lists</title>
  </head>

  <body>
    <h1>Markup Language Definitions</h1>
    <dl>
      <dt>SGML</dt>
        <dd>The Standard Generalized Markup Language</dd>
      <dt>HTML</dt>
         <dd>The Hypertext Markup Language</dd>
         <dd>The markup language you use to create Web pages.</dd>
      <dt>XML</dt>
        <dd>The Extensible Markup Language</dd>
    </dl>
  </body>
</html>
```

If you think the items in a list are spaced too closely together, you can either

- Put two `<br/>` elements before each `</li>` or `</dd>` element to add more white space.
- Use CSS styles to carefully control all aspects of your list appearance, as shown in Chapter 8.

Nesting lists

You can create subcategories by *nesting* lists within other lists. Some common uses for nested lists include

- Site maps and other navigation tools
- Table of contents for online books and papers
- Outlines

You can combine any of the three kinds of lists to create *nested* lists, such as a multilevel table of contents or an outline that mixes numbered headings with bulleted list items as the lowest outline level.

The following example starts with a numbered list that defines a list of things to do for the day, and uses three bulleted lists to further break down those items into specific tasks:

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

```
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Nested Lists</title>
  </head>
  <body>
    <h1>Things to do today</h1>
    <ol>
      <li>Feed cat</li>
        <ul>
          <li>Rinse bowl</li>
          <li>Open cat food</li>
          <li>Mix dry and wet food in bowl</li>
          <li>Deliver on a silver platter to fluffy</li>
        </ul>
      <li>Wash car</li>
        <ul>
          <li>Vacuum interior</li>
          <li>Wash exterior</li>
          <li>Wax exterior</li>
        </ul>
      <li>Grocery shopping</li>
        <ul>
          <li>Plan meals</li>
          <li>Clean out fridge</li>
          <li>Make list</li>
          <li>Go to store</li>
        </ul>
    </ol>
  </body>
</html>
```

All nested lists follows the same markup pattern:

- Each list item in the top-level ordered list is followed by a complete second-level list.
- The second-level lists sit inside the top-level list, not in the list items.

Figure 5-16 shows how a browser reflects this nesting in its display.

As you build nested lists, watch your opening and closing tags carefully. *Close first what you opened last* is an especially important axiom here. If you don't open and close your tags properly, lists might not show consistent indents or numbering, or text might be indented incorrectly because a list somewhere was never properly closed.

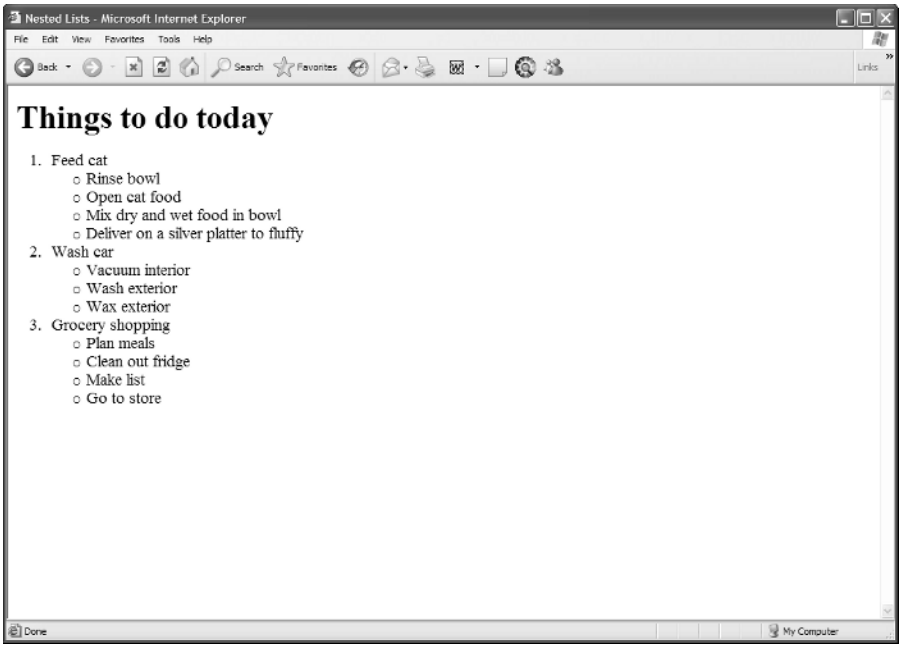

Figure 5-16: Nested lists combine lists for a multilevel organization of information.

Text Controls and Annotation

Some general (X)HTML elements define general text controls or allow you to annotate documents. These are covered in Table 5-1.

Table 5-1 (X)HTML Text Controls and Annotation

Element	*Common Name*	*Empty?*	*Category*	*Description*
`bdo`	Bidirectional algorithm	No	Language definition	Controls direction of text display {`ltr`|`rtl`} (left-to-right, right-to-left)
`del`	Deleted text	No	Text control	Marks deleted text in current draft
`ins`	Inserted text	No	Text control	Marks inserted text in current draft

Element	*Common Name*	*Empty?*	*Category*	*Description*
kbd	Keyboard text	No	Text control	Text to type at a keyboard
samp	Sample text	No	Text control	Sample program output
tt	Teletype text	No	Text control	Typewriter or teletype output
var	Variable text	No	Text control	Highlights input or output variables

Marvelous Miscellany

Table 5-2 lists other text-related (X)HTML attributes that you might find in HTML files.

Table 5-2 Additional (X)HTML Text Attributes

Name	*Function/ Value Equals*	*Value Type(s)*	*Related Element(s)*
cite	Specifies location of source materials	URL	<blockquote> <q>
cite	Explains reason for adds, deletes	Text	<del><ins>
datetime	Time stamps document content	ISO date	<del><ins>
dir	Specifies text direction for content	{ltr|rtl}	All elements except <base> <frame /><frameset> <iframe><param /> <script>
id	Supplies unique identifier for markup instances	ID	All elements except <base /><head> <html><meta /> <param /> <script><style> <title>

(continued)

Table 5-2 ***(continued)***

Name	*Function/ Value Equals*	*Value Type(s)*	*Related Element(s)*
`lang`	Names content language used	Language code	All elements except `<base />  <frame /><frameset> <iframe><param /> <script>`
`title`	Associates advisory info to content	Text	All elements except `<base /><head> <html><meta /> <param /><script> <style><title>`