TRAINING & REFERENCE

murach's HTML5 and CSS3

(Chapter 5)

Thanks for downloading this chapter from ***Murach's HTML5 and CSS3***. We hope it will show you how easy it is to learn from any Murach book, with its paired-pages presentation, its "how-to" headings, its practical coding examples, and its clear, concise style.

To view the full table of contents for this book, you can go to our **web site**. From there, you can read more about this book, you can find out about any additional downloads that are available, and you can review our other books on Java development.

Thanks for your interest in our books!

MIKE MURACH & ASSOCIATES, INC.
1-800-221-5528 • (559) 440-9071 • Fax: (559) 440-0963
murachbooks@murach.com • **www.murach.com**

What developers said about the previous edition of our HTML and CSS book

"If you want to get up to speed with CSS and HTML, this is a must-read in my opinion. I really liked how it gave lots of examples and had exercises at the end of each chapter. I never understood CSS positioning 100% until I read this book. Great job!"

Posted at an online bookseller

"This is the first book you will pick up when you want to figure out how to perform some aspect of web development."

Robert Schaffer, Houston Area League of PC Users

"As a 10-year veteran in web development, the more I read, I realized the book was explaining many things that I either have been taking for granted or have forgotten."

Posted at an online bookseller

"I started my career in IT before web browsers, web pages, and application servers existed. I often searched the web for information, but I was never able to find anything that helped me truly understand HTML. But after reading Murach's book, I can now say I understand HTML."

Eric Mortensen, Northeast Ohio Oracle Users Group

"What I like about this book is how it introduces CSS at the beginning, as a natural partner to HTML."

Larry Blake, vbCity.com

"I really like how the content is organized. Each topic is presented with an overview, practical advice, and exercises to reinforce the learning process. The technical explanation of each topic is concise, practical, and has sufficient depth. This book is perfect for anyone who would like to quickly learn HTML and CSS. It absolutely achieves its goals for training and reference."

Posted at an online bookseller

5

How to use the CSS box model for spacing, borders, and backgrounds

In the last chapter, you learned some basic CSS properties for formatting text. Now, you'll learn the properties for controlling the spacing between elements and for displaying borders and backgrounds. Specifically, you'll learn how to use the CSS box model for those purposes.

An introduction to the box model

When a browser displays a web page, it places each HTML block element in a box. That makes it easy to control the spacing, borders, and other formatting for elements like headers, sections, footers, headings, and paragraphs. Some inline elements like images are placed in a box as well. To work with boxes, you use the CSS *box model*.

How the box model works

Figure 5-1 presents a diagram that shows how the box model works. By default, the box for a block element is as wide as the block that contains it and as tall as it needs to be based on its content. However, you can explicitly specify the size of the content area for a block element by using the height and width properties. You can also use other properties to set the borders, margins, and padding for a block element.

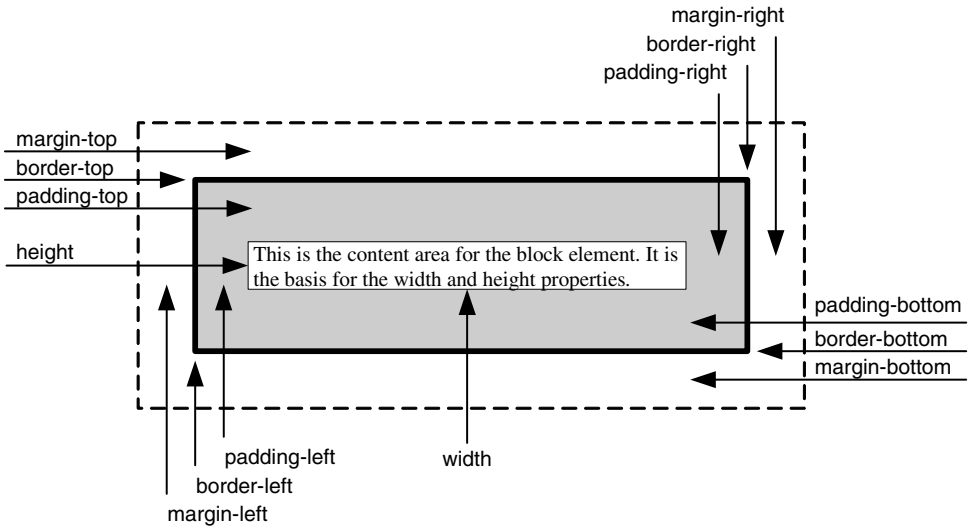
If you look at the diagram in this figure, you can see that *padding* is the space between the content area and a border. Similarly, a *margin* is the space between the border and the outside of the box.

If you need to calculate the overall height of a box, you can use the formula in this figure. Here, you start by adding the values for the margin, border width, and padding for the top of the box. Then, you add the height of the content area. Last, you add the values for the padding, border width, and margin for the bottom of the box. The formula for calculating the overall width of a box is similar.

When you set the height and width properties for a block element, you can use any of the units that you learned about in the last chapter. In most cases, though, you'll use pixels so the sizes are fixed. That way, the size of the page won't change if the user changes the size of the browser window. This is referred to as a *fixed layout*.

When you use a fixed layout, you can use either absolute or relative units of measure for margins and padding. If you use a relative unit such as ems, the margins and padding will be adjusted if the font size changes. If you use an absolute unit, the margins and padding will stay the same.

The CSS box model



The formula for calculating the height of a box

```
top margin + top border + top padding +
height +
bottom padding + bottom border + bottom margin
```

The formula for calculating the width of a box

```
left margin + left border + left padding +
width +
right padding + right border + right margin
```

Description

- The CSS *box model* lets you work with the boxes that a browser places around each block element as well as some inline elements. This lets you add formatting such as margins, padding, and borders.
- By default, the box for a block element is as wide as the block that contains it and as tall as it needs to be based on its content.
- You can use the height and width properties to specify the size of the content area for a block element explicitly.
- You can use other properties to control the margins, padding, and borders for a block element. Then, these properties are added to the height and width of the content area to determine the height and width of the box.

Figure 5-1 How the box model works

A web page that illustrates the box model

To help you understand how the box model works, figure 5-2 presents the HTML for a simple web page. Then, the CSS adds borders to the four types of elements in the HTML: a dotted 3-pixel border to the body, a solid 2-pixel border to the section, and dashed 1-pixel borders to the h1 and <p> elements. If you look at the web page in the browser, you can see how these four borders are rendered. You can also see how the margins and padding for these boxes work.

For the body, the margin on all four sides is set to 10 pixels. You can see that margin on the left, top, and right of the body border, but not on the bottom. That's because the bottom margin for the body is determined by the size of the window.

For the section, the width is set to 500 pixels, and the margins on all four sides of the box are set to 20 pixels. You can see these margins on the left, top, and bottom of the section box, but not on the right because the width of the section is set to 500 pixels.

The next rule set sets properties for both the h1 and <p> elements. In this case, the properties set the border and the padding for these elements. Then, the next two rule sets set additional properties for each of these elements.

The rule set for the h1 element sets the top margin to .5em, the right and left margins to 0, and the bottom margin to .25em. As a result, there is more space above the h1 element than below it. This rule set also sets the padding on the left side of the element to 15 pixels so space is added between the border of the box and the text.

The rule set for the <p> element starts by setting all the margins to 0. As a result, all of the space between the h1 and <p> elements is due to the bottom margin of the h1 element. In addition, the padding on the left side of the element is set to 15 pixels so the text for the h1 and <p> elements is aligned.

Please note that if I had used relative measures for the padding on the left of the h1 and <p> elements, they wouldn't be aligned because the font sizes for these elements are different. One more thing to notice is that the padding-left properties in the rule sets for the h1 and <p> elements override the left padding specified by the rule set for both of these elements.

This should help you understand how the box model works. Now, it's on to the details for setting the properties for the box model.

The HTML for a page that uses the box model

```
<body>
    <section>
        <h1>San Joaquin Valley Town Hall</h1>
        <p>Welcome to San Joaquin Valley Town Hall.
        We have some fascinating speakers for you this season!</p>
    </section>
</body>
```

The CSS for the page

```
body {
    border: 3px dotted black;
    margin: 10px;
}
section {
    border:  2px solid black;
    width:   500px;
    margin:  20px;          /* all four sides */
    padding: 10px;          /* all four sides */
}
h1, p {
    border: 1px dashed black;
    padding: 10px;
}
h1 {
    margin: .5em 0 .25em;   /* .5em top, 0 right and left, .25em bottom */
    padding-left: 15px;
}
p {
    margin: 0;              /* all four sides */
    padding-left: 15px;
}
```

The web page in a browser

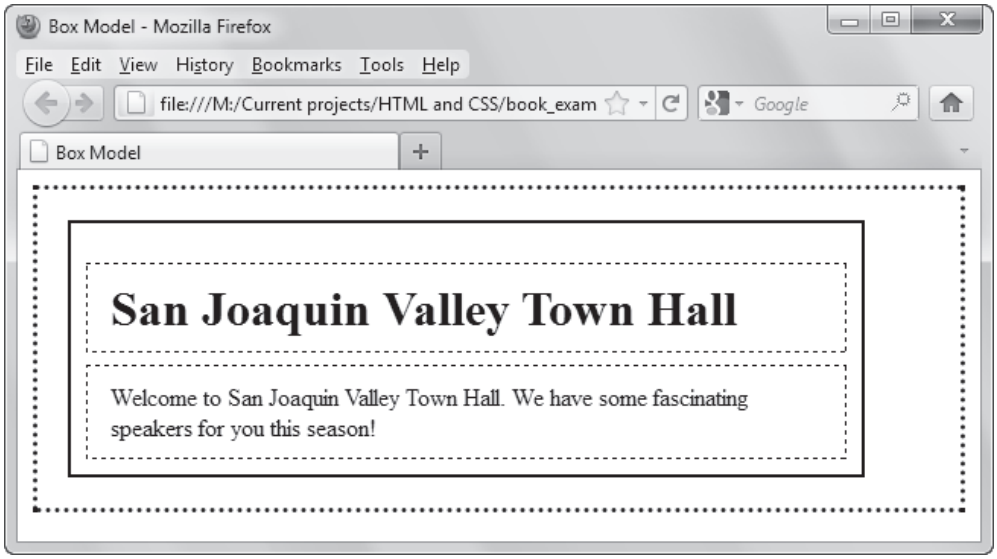


Figure 5-2 A web page that illustrates the box model

How to size and space elements

As you saw in the last figure, you can use several different properties to determine the size of an element and the spacing between the elements on a page. In the topics that follow, you'll learn the details of coding these properties.

How to set heights and widths

Figure 5-3 presents the properties for setting heights and widths. The two properties you'll use most often are width and height. By default, these properties are set to a value of "auto". As a result, the size of the content area for the element is automatically adjusted so it's as wide as the element that contains it and as tall as the content it contains. To change that, you can use the height and width properties.

The first two sets of examples in this figure illustrate how this works. Here, the first example in each set specifies an absolute value using pixels. Then, the second example specifies a relative value using percents. As a result, the width is set to 75% of the *containing block*.

Finally, the third example in each set uses the keyword "auto". That sets the width based on the size of the containing element and the height based on the content of the element. Because the default width and height is "auto", you usually won't need to use that value.

In addition to the width and height properties, you can use the min-width, max-width, min-height, and max-height properties to specify the minimum and maximum width and height of the content area. Like the width and height properties, you can specify either a relative or an absolute value for these properties. For instance, the last set of examples in this figure sets the values of these properties using pixels.

Properties for setting heights and widths

Property	Description
`width`	A relative or absolute value that specifies the width of the content area for a block element. You can also specify auto if you want the width of the box calculated for you based on the width of its containing block. This is the default.
`height`	A relative or absolute value that specifies the height of the content area for a block element. You can also specify auto if you want the height of the area calculated for you based on its content. This is the default.
`min-width`	A relative or absolute value that specifies the minimum width of the content area for a block element. The area will always be at least this wide regardless of its content.
`max-width`	A relative or absolute value that specifies the maximum width of the content area for a block element. You can also specify none to indicate that there is no maximum width.
`min-height`	A relative or absolute value that specifies the minimum height of the content area for a block element. The area will always be at least this tall regardless of its content.
`max-height`	A relative or absolute value that specifies the maximum height of the content area for a block element. You can also specify none to indicate that there is no maximum height.

How to set the width of the content area

```
width: 450px;          /* an absolute width */
width: 75%;            /* a relative width */
width: auto;           /* width based on its containing block (the default) */
```

How to set the height of the content area

```
height: 125px;
height: 50%;
height: auto;          /* height based on its content (the default) */
```

How to set the minimum and maximum width and height

```
min-width: 450px;
max-width: 600px;
min-height: 120px;
max-height: 160px;
```

Description

- If you specify a percent for the width property, the width of the content area for the block element is based on the width of the block that contains it, called the *containing block*. In that case, the width of the containing block must be specified explicitly.
- If you specify a percent for the height property, the height of the content area for the block element is based on the height of the containing block. In that case, the height of the containing block must be specified explicitly. Otherwise, "auto" is substituted for the percent.
- The min-width, max-width, min-height, and max-height properties are typically used to accommodate a change in font size.

Figure 5-3 How to set heights and widths

How to set margins

Figure 5-4 presents the properties for setting margins. As you can see, you can use individual properties like margin-top or margin-left to set individual margins. This is illustrated in the first set of examples in this figure.

Instead of setting individual margins, though, you can use the margin property to set the margins for all four sides of a box. When you use a *shorthand property* like this, you can specify one, two, three, or four values. If you specify all four values, they are applied to the sides of the box in a clockwise order: top, right, bottom, and left. To remember this order, you can think of the word *tr*ou*bl*e.

If you specify fewer than four values, this property still sets the margins for all four sides of the box. If, for example, you only specify one value, each margin is set to that value. If you specify two values, the top and bottom margins are set to the first value, and the left and right margins are set to the second value. And if you specify three values, the top margin is set to the first value, the left and right margins are set to the second value, and the bottom margin is set to the third value. This is illustrated in the second set of examples in this figure.

Although it isn't shown here, you can also specify the keyword "auto" for any margin. In most cases, you'll use this keyword to center a page in the browser window or a block element within its containing block. To do that, you specify auto for both the left and right margins. For this to work, you must also set the width of the element. You'll see an example of how this works in figure 5-6.

One more thing you should know about margins is that different browsers have different default margins for the block elements. Because of that, it's a good practice to explicitly set the top and bottom margins of the elements that you're using. That way, you can control the space between elements like headings and paragraphs.

Finally, if you specify a bottom margin for one element and a top margin for the element that follows it, the margins are *collapsed*. That means the smaller margin is ignored, and only the larger margin is applied. In that case, there may not be as much space between the two elements as you expected. One solution to this problem is to set the margins to zero and use padding for the spacing.

Properties for setting margins

Property	Description
`margin`	One to four relative or absolute values that specify the size of the margins for a box. One value is applied to all four margins. Two values are applied to the top and bottom and right and left margins. Three values are applied to the top, right and left, and bottom margins. And four values are applied to the top, right, bottom, and left margins (think *tr*ou*bl*e).
`margin-top`	A relative or absolute value that defines the space between the top border of an element and the top of the containing block or the bottom of the element above it.
`margin-right`	A relative or absolute value that defines the space between the right border of an element and the right side of the containing block or the left side of the element to its right.
`margin-bottom`	A relative or absolute value that defines the space between the bottom border of an element and the bottom of the containing block or the top of the element below it.
`margin-left`	A relative or absolute value that defines the space between the left border of an element and the left side of the containing block or the right side of the element to its left.

How to set the margin on a single side of an element

```
margin-top: .5em;
margin-right: 1em;
margin-bottom: 2em;
margin-left: 1em;
```

How to set the margins on multiple sides of an element

```
margin: 1em;               /* all four sides */
margin: 0 1em;             /* top and bottom 0, right and left 1em */
margin: .5em 1em 2em;      /* top .5em, right and left 1em, bottom 2em */
margin: .5em 1em 2em 1em; /* top .5em, right 1em, bottom 2em, left 1em */
```

Description

- If you specify a bottom margin for one element and a top margin for the element that follows in the HTML, the margins are *collapsed*, which means that only the larger margin is applied.
- You typically use the "auto" keyword to center an element in its *containing block*. To do that, you must also specify the width of the element.
- Because different browsers have different default margins for block elements, you often need to set these margins explicitly.

Figure 5-4 How to set margins

How to set padding

The properties for setting padding are similar to the properties for setting margins. These properties are presented in figure 5-5. As you can see, you can set the padding for the sides of a box individually, or you can set the padding for all four sides of a box at once using a shorthand property.

Properties for setting padding

Property	Description
`padding-top`	A relative or absolute value that defines the space between the top of an element and its top border.
`padding-right`	A relative or absolute value that defines the space between the right side of an element and its right border.
`padding-bottom`	A relative or absolute value that defines the space between the bottom of an element and its bottom border.
`padding-left`	A relative or absolute value that defines the space between the left side of an element and its left border.
`padding`	One to four relative or absolute values that specify the padding on multiple sides of an element. One value is applied to all four sides. Two values are applied to the top and bottom and right and left. Three values are applied to the top, right and left, and bottom. And four values are applied to the top, right, bottom, and left (think *tro*u*bl*e).

How to set the padding on a single side of an element

```
padding-top: 0;
padding-right: 1em;
padding-bottom: .5em;
padding-left: 1em;
```

How to set the padding on multiple sides of an element

```
padding: 1em;              /* all four sides */
padding: 0 1em;            /* top and bottom 0, right and left 1em */
padding: 0 1em .5em;       /* top 0em, right and left 1em, bottom .5em */
padding: 0 1em .5em 1em;   /* top 0em, right 1em, bottom .5em, left 1em */
```

Description

- If you set the top and bottom margins for elements to zero, you can use padding to set the spacing between the elements.

Figure 5-5 How to set padding

A web page that illustrates sizing and spacing

To illustrate the use of the properties for sizing and spacing, figure 5-6 presents a web page that uses many of these properties. This web page is similar to the one at the end of the last chapter. However, the page has been centered in the browser window, the spacing between the elements has been improved, and the second and third paragraphs aren't indented.

The HTML for the web page

The HTML for this web page is the same as in the application at the end of the last chapter, with two exceptions. First, the last paragraph in the section has an id attribute of "contact_us". Second, the second and third paragraphs in the section don't have a class attribute that's used to indent them.

If you want to review this HTML before you look at the CSS for this page, you can refer back to the last chapter. Or, you can open this page in your text editor. You shouldn't need to do that, though, because the focus here is on margins, padding, and the use of the box model.

A web page that uses widths, margins, and padding

Description

- This page is centered in the browser window and uses margins and padding to improve the spacing before and after headings, between list items, and between paragraphs.

Figure 5-6 A web page that illustrates the use of margins and padding

The CSS for the web page

Figure 5-7 presents the CSS for the web page. Here, I've highlighted all the style rules that affect the size and spacing of the elements on the page.

To start, the rule set for the body sets the width of the page to 600 pixels. Then, the top and bottom margins are set to 1em so there's space between the body and the top of the browser window. In addition, the left and right margins are set to "auto". Because a width is specified for the body, this causes the left and right margins to be calculated automatically so the page is centered in the browser window.

The next highlighted rule set is for the h1, h2, and <p> elements. This sets the margins and padding for these elements to 0. This prevents margin collapse when these items are next to each other. Then, the rule sets that follow can provide the right margins and padding for these elements.

For instance, the rule set for the <p> element sets the top and bottom padding to .25em, which provides the spacing between the paragraphs. Also, the rule set for the h1 element in the header sets the margin bottom to .25em, and the rule set for the h1 element in the section sets the margin top to 1em and the margin bottom to .35em. This provides the proper spacing before and after the headings.

In the rule set for the ul element, you can see that the bottom margin is set to 1.5em, and the other margins are set to zero. This reduces the space before the unordered list and increases the space after. Similarly, the bottom padding for the li elements is set to .35em. That provides the spacing after the list items, which means that you don't have to set the line height for the list.

Last, in the rule set for the element with "contact_us" as its id, you can see that the top margin is set to 1em, which provides the spacing before it. And the rule set for the footer sets the top margin to 1em which provides more space between it and the last paragraph in the section.

If you study all of this code, you can see that it avoids the problems of collapsing margins. It does that by first setting the margins for the elements to zero and then overriding those margins or using padding to provide the spacing before and after elements.

The CSS for the web page

```
/* So the HTML5 semantic elements are formatted in older browsers */
article, aside, figure, footer, header, nav, section {
    display: block;
}
/* the styles for the elements */
body {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 87.5%;
    width: 600px;
    margin: 1em auto; }       /* centers the page in the browser window */

h1 { font-size: 250%; }
h2 { font-size: 150%; }

h1, h2, p {                   /* sets the margins and padding to zero */
    margin: 0;
    padding: 0; }

a { font-weight: bold; }
a:link { color: green; }
a:hover, a:focus { color: blue; }

ul { margin: 0 0 1.5em; }
li { padding-bottom: .35em; }

p { padding: .25em 0; }
em { font-weight: bold; }

/* the styles for the header */
header img { float: left; }
header h1 {
    color: #ef9c00;
    text-align: center;
    text-shadow: 2px 2px 0 black;
    margin-bottom: .25em; }
header h2 {
    color: green;
    font-style: italic;
    text-align: center; }

/* the styles for the section */
section { clear: left; }
section h1 {
    font-size: 200%;
    margin: 1em 0 .35em; }
section a.date_passed  { color: gray; }
#contact_us { margin-top: 1em; }   /* adds space before the paragraph */

/* the styles for the footer */
footer { margin-top: 1em; }
footer p {
    font-size: 80%;
    text-align: right; }
```

Figure 5-7 The CSS for the web page

A version of the CSS that uses a reset selector

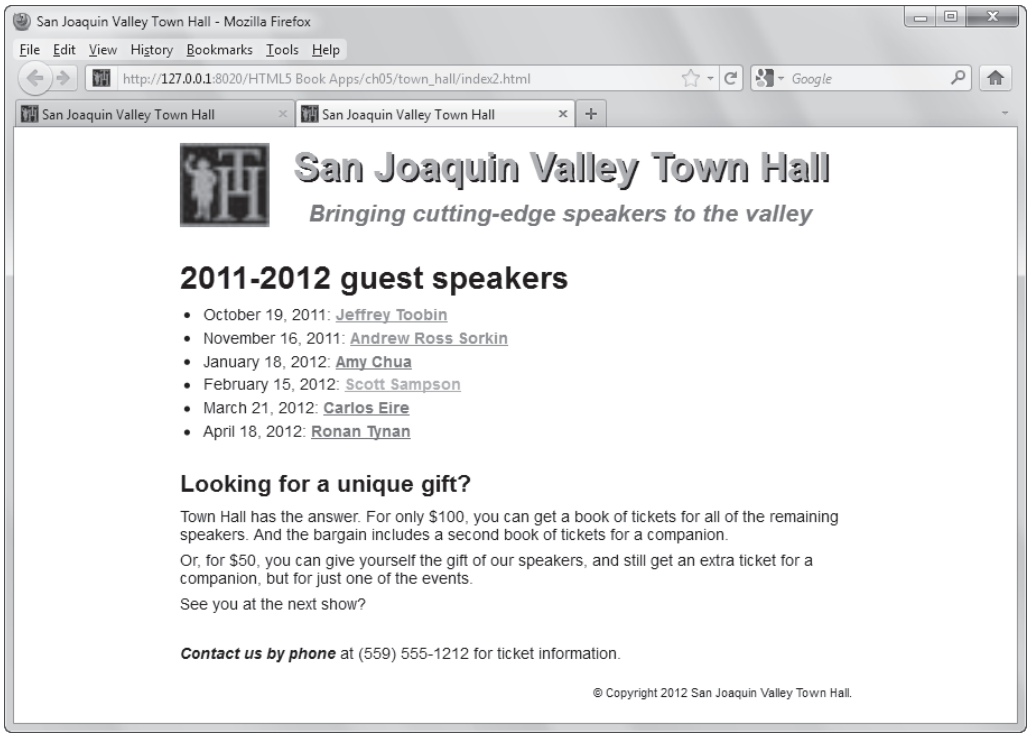
In figure 5-8, you can see another version of the CSS for this application. This time, the CSS uses a *reset selector*. That refers to the use of a universal selector that sets the margins and padding for all of the elements to zero. After you code the reset selector, you can get the spacing that you want by applying margins and padding to specific elements, which overrides the settings of the reset selector. This is an approach that's used by many professionals.

In the CSS for this figure, only the rule sets that provide margins or padding are listed. This works pretty much the way the code in the previous figure works, except for the ul and li elements. Because the reset selector set the margins and padding for these elements to zero, you need to provide margins and padding for these elements.

To bring the ul element in line with the heading above it, you need to set its left margin. In this example, the left margin is set to 1.25em. You also need to provide left padding for the li elements to provide space between the bullets and the text. In this example, that space is set to .25em.

Because of these CSS differences, the formatting of the web page in this figure is slightly different than the formatting in figure 5-6. Specifically, the bulleted list isn't indented, and there's more space between the bullets and text in the list items. Of course, you could make them the same by changing the settings for the ul and li elements, but this shows the options that you have when you use a reset selector. In chapter 7, you'll learn more about formatting lists.

A slightly modified version of the page that uses a reset selector



The CSS for this version of the page

```
/* the reset selector */
* {
    margin: 0;
    padding: 0; }

/* just the styles for the elements that provide margins or padding */
body {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 87.5%;
    width: 600px;
    margin: 1em auto; }

h1, h2 { margin-bottom: .25em; }

ul { margin: 0 0 1.5em 1.25em; }
li {
    padding-bottom: .35em;
    padding-left: .25em; }
p { padding: .25em 0; }

section h1 {
    font-size: 200%;
    margin-top: 1em; }
#contact_us { margin-top: 1.5em; }
footer { margin-top: 1em; }
```

Figure 5-8 A version of the CSS that uses a reset selector

How to set borders and backgrounds

Now that you know how to size and space elements using the box model, you're ready to learn how to apply other formatting to boxes. That includes adding borders and setting background colors and images.

How to set borders

Figure 5-9 presents the properties for setting *borders* and illustrates how they work. To start, if you want the same border on all four sides of a box, you can use the shorthand border property. This property lets you set the width, style, and color for the borders. This is illustrated in the first set of examples in this figure.

Here, the first rule creates a thin, solid, green border. Note that different browsers interpret the thin keyword, as well as the other keywords for width, differently. Because of that, you'll typically set the width of a border to an absolute value as shown in the second and third rules. Notice that the third rule doesn't specify a color. In that case, the border will be the same color as the element's text.

To set the border for just one side of a box, you can use the shorthand property for a border side. This is illustrated by the second set of examples. Here, the first rule sets the top border and the second rule sets the right border.

Most of the time, you'll use the shorthand properties to set all four borders or just one border of a box. However, you can use the other properties in this figure to set the width, styles, and colors of the sides of a border. This is illustrated by the last four groups of examples in this figure. You may want to use these properties to override another border setting for an element.

For instance, to set the width for each side of a border, you can use the border-width property as shown in the third set of examples. Here, you can specify one, two, three, or four values. This works just like the shorthand margin and padding properties you learned about earlier in this chapter. That means that the values are applied to the top, right, bottom, and left sides of the border.

The border-style and border-color properties are similar. You can use them to set the style and color for each side of a border by specifying one to four values. This is illustrated in the fourth and fifth sets of examples in this figure. Notice in the last example for the border-style property that the left and right borders are set to "none". That way, only the top and bottom borders will be displayed.

Properties for setting borders

Property	Description
border	A border width, border style, and border color. The values are applied to all sides of the border.
border-*side*	Border width, style, and color values for the specified side of a border.
border-width	One to four relative or absolute values (excluding a percent) or keywords that specify the widths for each side of a border. Possible keywords are thin, medium, and thick.
border-style	One to four keywords that specify the styles for each side of a border. Possible values are dotted, dashed, solid, double, groove, ridge, inset, outset, and none. The default is none.
border-color	One to four color values or keywords that specify the color for each side of a border. The default is the color of the element.
border-*side*-width	A relative or absolute value (excluding a percent) or a keyword that specifies the width of the indicated side of a border.
border-*side*-style	A keyword that specifies the style for the indicated side of a border.
border-*side*-color	A color value or keyword that specifies the color of the indicated side of a border.

The syntax for the shorthand border and border-*side* properties

```
border: [width] [style] [color];
border-side: [width] [style] [color];
```

How to set border properties

```
border: thin solid green;
border: 2px dashed #808080;
border: 1px inset;           /* uses the element's color property */
```

How to set side borders

```
border-top: 2px solid black;
border-right: 4px double blue;
```

How to set the widths of borders

```
border-width: 1px;              /* all four sides */
border-width: 1px 2px;          /* top and bottom 1px, right and left 2px */
border-width: 1px 2px 2px;      /* top 1px, right and left 2px, bottom 2px */
border-width: 1px 2px 2px 3px; /* top 1px, right 2px, bottom 2px, left 3px */
```

How to set the style of borders

```
border-style: dashed;     /* dashed line all sides */
border-style: solid none; /* solid top and bottom, no border right and left */
```

How to set the color of borders

```
border-color: #808080;
border-color: black gray; /* black top and bottom, gray right and left */
```

How to set the width, style, and color of border sides

```
border-bottom-width: 4px;
border-right-style: dashed;
border-left-color: gray;
```

Figure 5-9 How to set borders

How to use CSS3 to add rounded corners and shadows to borders

Figure 5-10 shows how to use the CSS3 features for adding *rounded corners* and *shadows* to borders. This lets you supply these graphic effects without using images. These features are currently supported by all modern browsers. But if an older browser doesn't support them, it just ignores them, which is usually okay. That's why you can start using these features right away.

To round the corners, you use the border-radius property that's summarized in this figure. If you supply one value for it, it applies to all four corners of the border. But if you supply four values as in the example, you can apply specific rounding to each corner. Here, the upper-left corner has a 10 pixel radius, the upper-right corner has a 20 pixel radius, the lower-right corner has a zero radius so it isn't rounded, and the lower-left corner has a 20 pixel radius.

To add shadows to a border, you use the box-shadow property. This works much like the text-shadow property that you learned about in the last chapter. With the first two values, you specify the offset for the shadow. With the third value, you can specify the blur radius. With the fourth value, you specify how far the blur is spread. And with the fifth value, you can specify a different color for the shadow than the one for the border. To get the effects that you want, you usually need to experiment with these values, but this is easier than using an image to get those effects.

Although all modern browsers support these features, you can also provide backward compatibility for older browsers by using the properties that are summarized in the compatibility guidelines in this figure. For instance, you can get these features to work in older Firefox browsers by using the border-radius and box-shadow properties with -moz- as the prefix. And you can get the box-shadow property to work in older Safari and Chrome browsers by using -webkit- as the prefix.

Incidentally, there are some options for these features that aren't presented in this figure. For instance, you can use a separate property like border-top-left-radius for each corner. You can also set the curvature of a corner by supplying two values for a single corner like this:

```
border-top-left-radius: 50px 20px;
```

So, if you want to go beyond what this figure offers, please refer to the W3C documentation.

The syntax for the border-radius and box-shadow properties

```
border-radius: radius;  /* applies to all four corners */
border-radius: topLeft topRight lowerRight lowerLeft;
box-shadow: horizontalOffset verticalOffset blurRadius spread color;
```

The HTML for a section

```
<section>
    <a href="ebooks_index.html">$10 Ebooks!</a>
</section>
```

The CSS for the section

```
section {
    padding: 20px;
    width: 160px;
    border: 5px double blue;
    color: blue;
    font-size: 200%;
    text-align: center;
    font-weight: bold;
    border-radius: 10px 20px 0 20px;
    box-shadow: 3px 3px 4px 4px red;
}
```

The section in a browser

Guidelines for backward compatibility

- To round corners and add shadows in older versions of Firefox, you can use the -moz-border-radius and -moz-box-shadow properties.
- To add shadows in older versions of Safari and Chrome, you can use the -webkit-box-shadow property.

Discussion

- When you code the border-radius property, you can assign one rounding radius to all four corners or a different radius to each corner.
- When you code the box-shadow property, positive values offset the shadow to the right or down, and negative values offset the shadow to the left or up.
- The third value in the box-shadow property determines how much the shadow is blurred, and the fourth value determines how far the blur is spread.
- The fifth value in the box-shadow property specifies the color of the shadow. If this is omitted, it is the same color as the border.
- These properties are supported by all modern browsers. If they aren't supported by a browser, they are ignored, which is usually okay.

Figure 5-10 How to use CSS3 to add rounded corners and shadows to borders

How to set background colors and images

Figure 5-11 presents the properties you can use to set the *background* for a box. When you set a background, it's displayed behind the content, padding, and border for the box, but it isn't displayed behind the margin.

When you specify a background, you can set a background color, a background image, or both. If you set both, the browser displays the background color behind the image. As a result, you can only see the background color if the image has areas that are transparent or the image doesn't repeat.

As this figure shows, you can set all five properties of a background by using the shorthand background property. When you use this property, you don't have to specify the individual properties in a specific order, but it usually makes sense to use the order that's shown. If you omit one or more properties, the browser uses their default values.

The first set of examples illustrates how this works. Here, the first rule sets the background color to blue, the second rule sets the background color and specifies the URL for an image, and the third rule specifies all five background properties. You can also code each of these properties individually.

By default, the background color for a box is transparent. As a result, you can see through the box to the color that's behind it, which is usually what you want. Otherwise, you need to set the color. The first rule in the second set of examples shows how to do that using the background-color property.

If you want to display a background image, you need to specify a URL that points to the file for the image. You can do that using the background-image property as shown in the second rule in the second set of examples.

If you add a background image to a box, it will repeat horizontally and vertically to fill the box by default. This works well for small images that are intended to be tiled across or down a box. If you want to change this behavior, you can set the background-repeat property so the image is only repeated horizontally, so it's only repeated vertically, or so it isn't repeated at all. This is illustrated by the first four rules in the last set of examples.

If an image doesn't repeat horizontally and vertically, you may need to set additional properties to determine where the image is positioned and whether it scrolls with the page. By default, an image is positioned in the top left corner of the box. To change that, you use the background-position property. This property lets you specify both a horizontal and a vertical position.

The next three rules illustrate how this works. The first rule positions the image at the top left corner of the box, which is the default. The second rule centers the image at the top of the box. And the third rule positions the image starting 90% of the way from the left side to the right side of the box and 90% of the way from the top to the bottom of the box.

In most cases, you'll want a background image to scroll as you scroll the box that contains it. For example, if you use a background image for an entire page and the page is larger than the browser window, you'll usually want the image to scroll as you scroll through the page. If not, you can set the background-attachment property to "fixed".

The properties for setting the background color and image

Property	Description
`background`	Background color, image, repeat, attachment, and position values.
`background-color`	A color value or keyword that specifies the color of an element's background. You can also specify the transparent keyword if you want elements behind the element to be visible. This is the default.
`background-image`	A relative or absolute URL that points to the image. You can also specify the keyword none if you don't want to display an image. This is the default.
`background-repeat`	A keyword that specifies if and how an image is repeated. Possible values are repeat, repeat-x, repeat-y, no-repeat. The default is repeat, which causes the image to be repeated both horizontally and vertically to fill the background.
`background-attachment`	A keyword that specifies whether an image scrolls with the document or remains in a fixed position. Possible values are scroll and fixed. The default is scroll.
`background-position`	One or two relative or absolute values or keywords that specify the initial horizontal and vertical positions of an image. Keywords are left, center, and right; top, center, and bottom. If a vertical position isn't specified, center is the default. If no position is specified, the default is to place the image at the top-left corner of the element.

The syntax for the shorthand background property

```
background: [color] [image] [repeat] [attachment] [position];
```

How to use the shorthand property

```
background: blue;
background: blue url("../images/texture.gif");
background: #808080 url("../images/header.jpg") repeat-y scroll center top;
```

How to set the background color and image with separate properties

```
background-color: blue;
background-image: url("../images/texture.gif");
```

How to control image repetition, position, and scrolling

```
background-repeat: repeat;          /* repeats both directions */
background-repeat: repeat-x;        /* repeats horizontally */
background-repeat: repeat-y;        /* repeats vertically */
background-repeat: no-repeat;       /* doesn't repeat */

background-position: left top;      /* 0% from left, 0% from top */
background-position: center top;    /* centered horizontally, 0% from top */
background-position: 90% 90%;       /* 90% from left, 90% from top */

background-attachment: scroll;      /* image moves as you scroll */
background-attachment: fixed;       /* image does not move as you scroll */
```

Accessibility guideline

- Don't use a background color or image that makes the text that's over it difficult to read.

Figure 5-11 How to set background colors and images

How to use CSS3 to set background gradients

Figure 5-12 shows the basics of how to use the new CSS3 feature for *linear gradients*. Here again, this feature lets you provide interesting backgrounds without using images. And if a browser doesn't support this feature, it just ignores it, which is usually okay.

At present, all modern browsers support this feature except IE9. However, the browsers support this feature with properties that are prefixed by -moz- for Firefox, -webkit- for Safari and Chrome, and -o- for Opera. Eventually, though, these prefixes will be dropped. For simplicity, the examples in this figure only use the –moz- prefix, but all of the prefixes are used in figure 5-15.

If you study the examples in this figure, you'll start to see how this feature works. In the first example, the color goes from left to right; the first color is white starting at the far left (0%), and the second color is red ending at the far right (100%). Then, CSS provides the gradient from left to right when the page is rendered.

In the second example, the direction is 45 degrees and there are three color groups. Red starts at the far left (0%). White is in the middle (50%). And blue is on the right (100%). Then, CSS provides the gradients when the page is rendered.

In the third coding example, after the browser display, you can see how to code solid stripes. Here, both a starting and ending point is given for each color. The result is three solid stripes of red, white, and blue at a 45 degree angle.

If you experiment with this feature, you'll see the interesting effects that you can get with it. But you can take this feature even further. For complete details, please refer to the W3C documentation.

The syntax for using a linear gradient in the background-image property

```
background-image:
    linear-gradient(direction, color %, color %, ... );
```

The HTML for two divisions

```
<div id="eg1"></div>
<div id="eg2"></div>
```

The CSS for the two divisions using Mozilla prefixes (-moz-)

```
#eg1 {
    background-image: -moz-linear-gradient(left, white 0%, red 100%); }
#eg2 {
    background-image: -moz-linear-gradient(
        45deg, red 0%, white 50%, blue 100%); }
```

The linear gradients in a browser

A background-image property that creates red, white, and blue stripes

```
background-image:
   -moz-linear-gradient(45deg, red 0%, red 33%, white 33%, white 66%,
                               blue 66%, blue 100%)
```

Discussion

- The CSS3 for *linear gradients* lets you create gradients for backgrounds without using images. That's why this feature will be useful when it's fully implemented.
- Today, all of the current browsers except IE9 support linear gradients. However, they implement them with their own prefixes: -moz- for Mozilla Firefox, -webkit- for WebKit browsers like Safari and Chrome, and -o- for Opera. Later, when the final CSS3 specification is set, these prefixes will be removed.
- The first parameter of a linear gradient indicates the direction the gradient will go: left for left to right, top for top to bottom, right for right to left, bottom for bottom to top, and a number of degrees if the gradient should be on an angle.
- The direction is followed by two or more parameters that consist of a color and a percent. The first percent indicates where the first color should start, the last percent indicates where the last color should end, and the percents in between indicate the points at which one gradient stops and the next one starts.

Figure 5-12 How to use CSS3 to set background gradients

A web page that uses borders and backgrounds

Figure 5-13 presents a web page that's similar to the one in figure 5-6. In fact, the HTML for these two pages is identical. However, the page in this figure has some additional formatting.

First, this page uses a gradient for the background behind the body. It also uses white as the background color for the body. That way, the gradient doesn't show behind the body content.

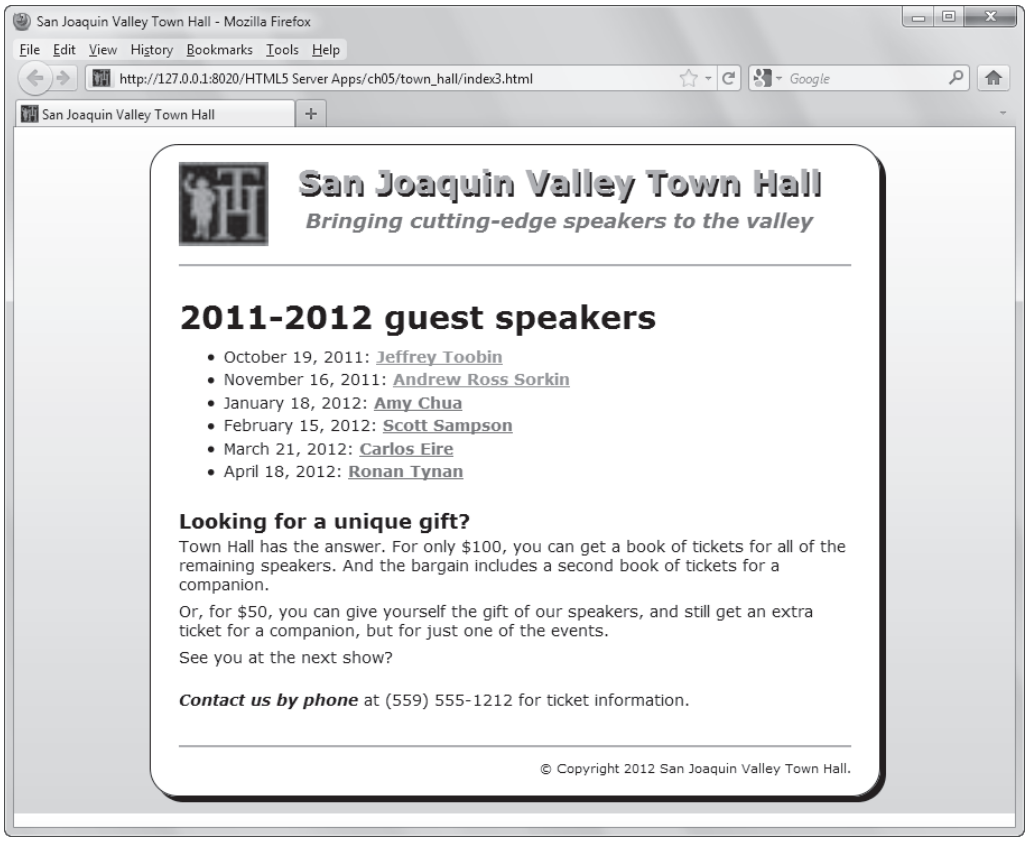
Second, this page has a border around the body with rounded corners and shadows. Also, the header has a border below it, and the footer has a border above it.

Third, the shadow for the text in the h1 element of the heading has been adjusted so the shadow is black and there is no blur. This makes this heading easier to read.

The HTML for the web page

The HTML is identical to the HTML for the page shown in figure 5-6. If you want to review this HTML, you can open the HTML for this application in your text editor.

A web page that uses borders and a background gradient



Description

- The HTML for this web page is the same as for figure 5-6.
- The styles for this web page include a border around the page content, a border below the header, and a border above the footer.
- The border around the body of the page has rounded corners and a drop shadow, and the background behind the body is a linear gradient. If you use a browser that doesn't support these CSS3 features, they will be ignored.

Figure 5-13 A web page that uses borders and a linear gradient for the background

The CSS for the web page

Figure 5-14 presents the CSS for the web page in figure 5-13. In this case, I've highlighted the rules that are different from the ones in figure 5-7. Although you should be able to understand this code without any help, here are some points that you should note.

First, a gradient is used for the background image of the html element. This is the gradient that you see on the top, bottom, and sides of the body element. To make this work in all browsers, the gradient is coded four times: one time each for the -moz-, -webkit-, and -o- prefixes, and one time without a prefix. When this rule set is rendered by a browser, the browser will ignore any rules that it doesn't recognize and render the first rule that it does recognize. When all of the browsers drop their prefixes, the last rule is the one that the browsers will execute.

Second, the background color of the body of the document is set to white. That way, the contents of the page will be displayed on a white background. Also, the top and bottom margins for the body are set to 15px, which lets the gradient show above and below the body, and the right and left margins are set to auto, which centers the body in the browser window. Next, the top and bottom padding for the body is set to 15px and the left and right padding is set to 25px. Then, a border with rounded corners and a shadow is added to the body.

Third, a border has been added below the header and above the footer. For the footer, the top margin is 2 ems, which is the space above the border, and top padding is .7 ems, which is the space between the border and the paragraph below it.

Last, the text-shadow property for the header h1 selector has been changed so there is no blur and the shadow is black. If you compare this heading in figure 5-13 with the one in figure 5-6, I think you'll agree that it's easier to read.

If you want the rounded corners and shadows to work in older browsers, you could add the prefixed properties that are described in the compatibility guidelines in figure 5-10. But if the body corners aren't rounded and shadowed, there will be little effect on the user experience. Also, the percentage of older browsers goes down every year so this is less of an issue each year. As a result, there is little to be gained by adding these properties to the code.

The CSS for the web page

```
/* the styles that provide the background behind the body */
html {
    background-image: -moz-linear-gradient(top, white 0%, #facd8a 100%);
    background-image: -webkit-linear-gradient(top, white 0%, #facd8a 100%);
    background-image: -o-linear-gradient(top, white 0%, #facd8a 100%);
    background-image: linear-gradient(top, white 0%, #facd8a 100%);
}
/* the styles for the body that is centered and bordered */
body {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    font-size: 87.5%;
    width: 600px;
    background-color: white;
    margin: 15px auto;
    padding: 15px 25px;
    border: 1px solid black;
    border-radius: 25px;
    box-shadow: 5px 5px 0 0;
}

/* the styles for the other type selectors are the same as in figure 5-7 */

/* the styles for the header */
header {
      padding-bottom: 2em;
      border-bottom: 2px solid #ef9c00; }
header img { float: left; }
header h1 {
    color: #ef9c00;
    text-align: center;
    text-shadow: 2px 3px 0 black;
    margin-bottom: .25em; }
header h2 {
    color: green;
    font-style: italic;
    text-align: center; }

/* the styles for the section */
section { clear: left; }
section h1 {
      font-size: 200%;
      margin: 1em 0 .35em; }
section a.date_passed  { color: gray; }
#contact_us { margin-top: 1em; }

/* the styles for the footer */
footer {
      margin-top: 2em;
      border-top: 2px solid #ef9c00;
      padding-top: .7em; }
footer p {
      font-size: 80%;
      text-align: right; }
```

Figure 5-14 The CSS for the web page

Perspective

Now that you've completed this chapter, you should understand how the box model is used for margins, padding, borders, backgrounds, and background images. As a result, you should be able to get the spacing, borders, and backgrounds for your web pages just the way you want them.

In the next chapter, though, you'll learn how to use CSS for laying out the elements on a page in two- and three-column arrangements with both headers and footers. That will complete your crash course in HTML and CSS.

Terms

box model
padding
margin
fixed layout
containing block
shorthand property
collapsed margins
reset selector
border
background
rounded corners
shadows
linear gradient

Summary

- The CSS *box model* refers to the box that a browser places around each block element as well as some inline elements. Each box includes the content of the element, plus optional padding, borders, and margins.
- To set the height and width of a content area, you can use absolute measurements like pixels or relative measurements like percents. If you use a percent, the percent applies to the block that contains the box you're formatting.
- You can set the *margins* for all four sides of a box. Because different browsers have different default values for margins, it's good to set the margins explicitly.
- If you specify a bottom margin for one element and a top margin for the element that follows, the margins are *collapsed* to the size of the largest margin.
- Like margins, you can set the *padding* for all four sides of a box. One way to avoid margin collapse is to set the margins to zero and use padding for the spacing.
- A *border* can be placed on any of the sides of a box. That border goes on the outside of the padding for the box and inside any margins, and you can set the width, style, and color for a border.
- When you set the *background* for a box, it is displayed behind the content, padding, and border for the box, but not behind the margins. The background can consist of a color, an image, or both.
- You can use CSS3 to *round corners* and add *shadows* to borders. You can also use CSS3 to provide *linear gradients* as backgrounds.

Exercise 5-1 Enhance the Town Hall home page

In this exercise, you'll enhance the formatting of the Town Hall home page that you formatted in exercise 4-1. When you're through, the page should look like this:

Open the HTML and CSS files and start enhancing the CSS

1. Use your text editor to open the HTML and CSS files that you created for exercise 4-1:

   ```
   c:\html5_css3\exercises\town_hall_1\index.html
   c:\html5_css3\exercises\town_hall_1\styles\main.css
   ```

2. Enhance the rule set for the body, by setting the width to 600 pixels, setting the top and bottom margins to 0 and the right and left margins to auto, and adding a 3-pixel, solid border with #800000 as its color. Then, test this change in Firefox. If the page isn't centered with a border, make the required corrections.
3. Add one more rule to the body that sets the background color to #fffded. Then, test this change, and note that the entire window is set to the background color, not just the body.
4. To fix this, code a rule set for the html element that sets the background color to white. Then, test to make sure that worked.

Add the other borders

From this point on, test each change in Firefox right after you make it.

5. Add a bottom border to the header and a top border to the footer. Both borders should be the same as the border around the body.
6. You should already have a rule set that uses an id selector to select this heading: "Our 2011-2012 Speakers". Find it, and delete any rules that it contains. Then, code rules that add top and bottom borders to this heading. Both borders should be the same as the borders for the header and footer.

Get the padding right for the header, section, and footer

At this point, you have all of the borders and colors the way they should be, so you just need to set the margins and padding. In the steps that follow, you'll start by adding a reset selector to the CSS file. Then, with one exception, you'll use padding to get the spacing right.

7. Add a reset selector like the one in figure 5-8 to the CSS file. When you test this change, the page won't look good at all because the default margins and padding for all of the elements have been removed.
8. For the header, add 1.5 ems of padding at the top and bottom. Then, delete the text-indent rules for the h1 and h2 elements in the header, and add 30 pixels of padding to the right and left of the image in the header. When you test these changes, and you'll see that the heading looks much better.
9. For the section, add 2 ems of padding to the right and left.

Get the padding right for the headings and text

10. In the section, set the padding for the headings and text as follows:

Element	Padding
h1	.5em top, .25em bottom
h2	.25em bottom
img	1em bottom
p	.5em bottom
blockquote	2em right and left
ul	1.5em bottom, 1.25em left
li	.35em bottom

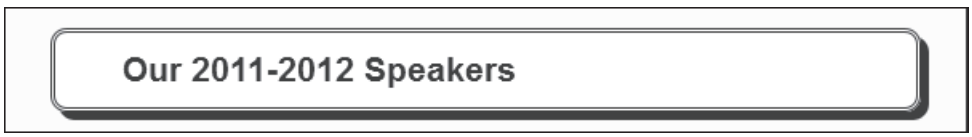
11. Set the padding for the top and bottom of the footer to 1em.
12. Test these changes in Firefox. At this point, everything should look right except that there isn't enough space between the bottom border of the 2011-2012 Speakers heading and the date that follows it. One way to fix this is to add a .75em bottom margin to the Speakers heading, which will add space below the bottom border. Try that.

Add the finishing touches and test in IE

13. Italicize the blockquote element to make it stand out.
14. Add a linear gradient as the background for the header. The one that's shown uses #800000, #fffded, white, #ffded, and #800000 as its five colors at a 30 degree angle. But experiment with this until you get it the way you want it.
15. Do one final test to make sure that the page looks like the one at the start of this exercise. Then, experiment on your own to see if you can improve the formatting. For instance, you may want to add some space between the author names and their images.
16. When you're through experimenting, test the page in IE. There the text shadow and the linear gradient may not work because they still aren't supported. But if you see any other problems, fix them and test again in both Firefox and IE.

Exercise 5-2 Add rounded corners and box shadows to the Speakers heading

Use CSS to add a double border with rounded corners and box shadows to the Speakers heading so it looks like this:



This should work in both Firefox and IE.

How to build your web development skills

Mike Murach, Publisher

The easiest way is to let **Murach's HTML5 and CSS3** be your guide! So if you've enjoyed this chapter, I hope you'll get your own copy of the book today. You can use it to:

- Teach yourself how to create web pages the way today's best web developers do, using HTML5 semantic tags to structure the page content and CSS to format and lay out the content on the page
- Add forms and audio or video clips to your sites, and format your pages so they're easy to print
- Enhance your web pages by using tested JavaScript and jQuery routines for features like image rollovers, slide shows, accordions, and carousels
- Pick up new skills whenever you want to or need to by focusing on material that's new to you
- Look up coding details or refresh your memory on forgotten details when you're in the middle of developing a web site
- Loan to your colleagues who are always asking you questions about web development

To get your copy, you can order online at **www.murach.com** or call us at 1-800-221-5528 (toll-free in the U.S. and Canada). And remember, when you order directly from us, this book comes with my personal guarantee:

100% Guarantee

You must be satisfied. Each book you buy directly from us must outperform any competing book or course you've ever tried, or send it back within 90 days for a full refund…no questions asked.

Thanks for your interest in Murach books!