Tutorial: Margins, Backgrounds, and Borders

In this tutorial, you'll explore elements of the CSS box model, adjust the spacing around objects on a page, add colorful borders to items on a page, and control the size and flow of page elements.

Controlling Page Margins and Backgrounds

You'll start with a very basic HTML file containing an internal style sheet with a basic CSS reset style. It's not much to look at right now (see Figure 7-18).

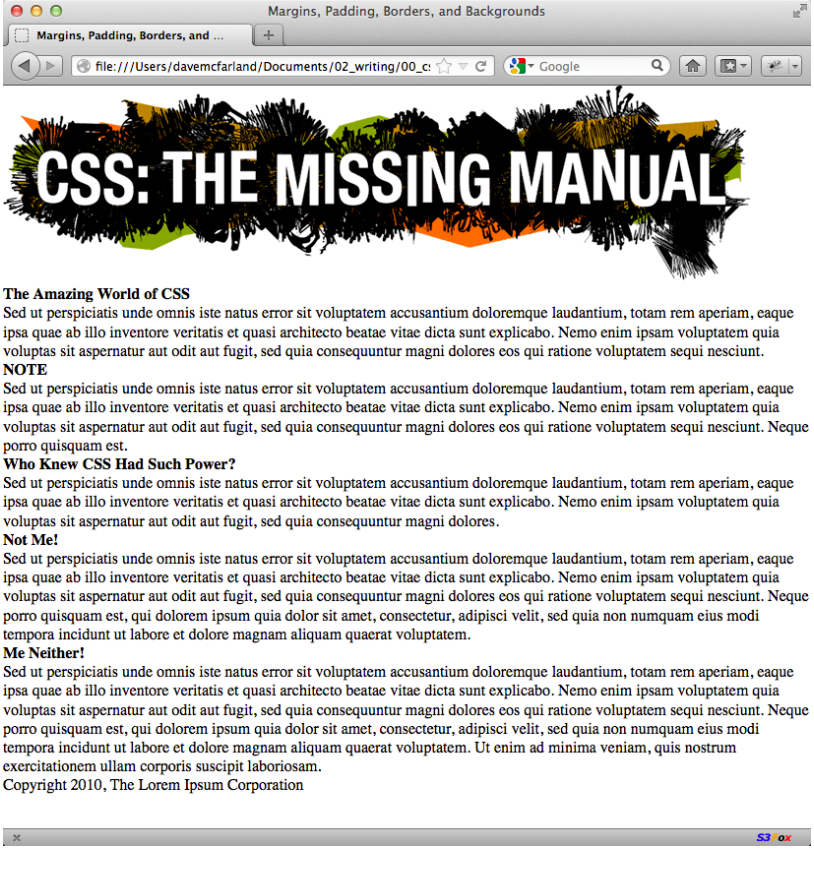

FIGURE 7-18
This web page is barebones HTML, with a single style that removes much of the built-in web browser styling. It'll look a lot better with a box model makeover.

NOTE For a sneak preview of the final result, check out Figure 7-21.

1. **In your favorite text editor, open *07→main.css*.**

   This style sheet is already linked to the *index.html* file, so the styles you add here will apply to that web page. The styles here (the same set of styles discussed on page 109) basically remove all margins, padding, and font size from the most common block-level elements and eliminate many of the cross-browser display problems you'll encounter related to these properties.

   Probably the most important properties are the `margin` and `padding` settings in the first style. There's enough cross-browser weirdness related to those two properties that many designers zero them out and start fresh. Another common alternative is a style sheet that eliminates cross-browser display differences, but still keeps some basic margins in place: *normalize.css* (*http://necolas.github.io/normalize.css/*) is one common choice.

   You'll start with something simple: a background color.

2. **At the bottom of the main.css file, click directly after the CSS comment** `/* end reset styles */` **and add a tag selector style:**

```
html {
 background-color: rgb(253,248,171);
}
```

   This style adds a light yellow background color to the page. If you want to color the background of a web page, you can add the `background-color` property to either the `<html>` tag or the `<body>` tag. Next, you'll add some margins, borders, and other properties to the `<body>` tag.

NOTE You may be used to using hexadecimal colors (like #FDF8AB) instead of RGB colors. You can use a tool like the online convertor at *www.colorhexa.com* to convert between the two. Using RGB is a good idea because RGBA colors, with their optional transparency (page 149), are so useful, and it's easier to just stick with one color model (RGB) instead of mixing two (RGB and hex).

3. **Add another style to the style sheet:**

```
body {
  background-color: rgb(255,255,255);
  border: 3px solid rgb(75,75,75);
}
```

   This style adds a white background color to the `<body>` tag and a 3-pixel dark gray border. Because the `<body>` tag sits inside the `<html>` tag, a web browser considers it to be "on top" of the `<html>` tag, so the white background will cover the yellow color you added in the previous step. Next you'll give the `<body>` tag a width and adjust its padding and margins.

TIP Normally, if you add a background color property to the <body> tag, that color fills the entire browser window; however, if you also add a background color to the <html> tag, the body's background color fills only the area that has content. To see this in action, just preview the web page after completing step 4 above; then delete the `html` tag style, and preview the page again. A weird, but useful, bit of CSS trivia.

4. **Edit the body style you just created by adding five new properties (changes are in bold):**

```
body {
  background-color: rgb(255,255,255);
  border: 3px solid rgb(75,75,75);
  max-width: 760px;
  margin-top: 20px;
  margin-left: auto;
  margin-right: auto;
  padding: 15px;
}
```

The `max-width` property constrains the body so that it never gets more than 760 pixels wide: If a visitor's browser window is wider than 760 pixels, then he'll see the background color from the `html` style and a 760-pixel box with the white background of the <body> tag. However, the browser window can get smaller than that, and the body will then shrink to fit the window, which makes viewing the page on a small tablet or phone easier.

The `margin-top` property adds 20 pixels of space from the browser window's top edge—nudging the <body> tag down just a bit—while the left and right margin settings center the body in the middle of the browser window. "Auto" is just another way of telling a browser, "You figure it out," and since that `auto` value is applied to both the left and right margins, a browser simply provides equal space on the left and right side.

NOTE You could also use the `margin` shorthand property (page 189) to condense those three lines of margin settings to just one, like this:

```
margin: 20px auto 0 auto;
```

Finally, to keep the content inside the <body> tag from touching the border line, 15 pixels of space are added to the inside of the body by using the `padding` property—in other words, the image and text are indented 15 pixels from all four edges. Next, you'll add a glow around the box using the `box-shadow` property.

5. **Edit the** body **style you just created by adding one last property after the border but before the width (changes are in bold):**

```
body {
  background-color: rgb(255,255,255);
  border: 3px solid rgb(75,75,75);
  box-shadow: 0 0 15px 5px rgba(44,82,100,.75);
  max-width: 760px;
  margin-top: 20px;
  margin-left: auto;
  margin-right: auto;
  padding: 15px;
}
```

   This style adds a glow to the box by creating a 15-pixel shadow placed directly behind the box (the `0 0` part at the beginning indicates that the shadow isn't offset to the left/right or top/bottom; it's simply in the background). The `5px` value is the spread value (page 203), and it pushes the shadow out 5 pixels around all four edges. Finally, the `rgba` value sets the color to a dark blue that's only 75 percent solid (that is, you can see through to the background yellow).

   Your style sheet is pretty far along, and you're ready to check the page.

6. **Save the file and preview the page in a web browser.**

   You should see a white box with an image, a bunch of text, and a gray outline with a bluish glow floating in a sea of yellow (see Figure 7-19). The text needs some loving attention. You'll take care of that next.

Adjusting the Space Around Tags

Since the CSS reset styles pretty much stripped the text on this page of all formatting, you'll need to create styles to make the headings and paragraphs look great. You'll start with the <h1> tag at the top of the page.

1. **Return to your text editor and the *main.css* file. Click at the end of the closing brace of the** <body> **tag selector, press Enter (Return) to create a new line, and then add the following style:**

```
h1 {
 font-size: 2.75em;
 font-family: Georgia, "Times New Roman", Times, serif;
 font-weight: normal;
 text-align: center;
 letter-spacing: 1px;
 color: rgb(133,161,16);
 text-transform: uppercase;
}
```

This style uses many of the text-formatting properties discussed in the previous chapter—the top headline is 2.75 ems tall (44 pixels in most browsers) and all uppercase, uses the Georgia font, and has a green color, with a little space between each letter. The `text-align` property makes sure the text is centered in the middle of the box. The real fun is adding a background color to really highlight the headline.

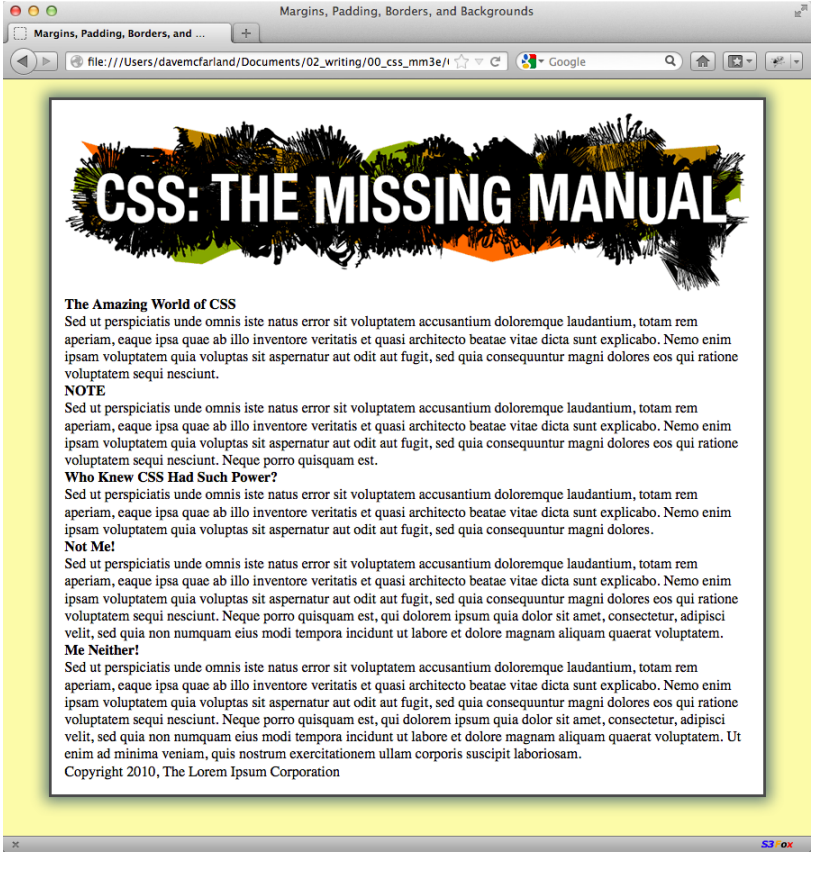

FIGURE 7-19

Setting the left and right margins to `auto` *for any element with a set width centers it. In this case, setting a width for the body and adding* `margin-left: auto;` *and* `margin-right: auto;` *places it smack dab in the center of the browser window. Unfortunately, to center an element vertically (with equal space above and below it) you need to use something more than margin properties. For a great overview of vertical (and horizontal) centering tricks, check out https://css-tricks.com/centering-css-complete-guide/.*

TIP Save the file and preview it in a web browser after each step in this tutorial. That way, you'll get a better understanding of how these CSS properties affect the elements they format. You may need to press the Ctrl or ⌘ key as you reload the web page to force the browser to reload the *main.css* file and not use the version that the browser stored in its cache. (See page 25 for more on how a browser's cache works).

2. **Add one new property to the `h1` tag style so that it looks like this (changes in bold):**

```
h1 {
 font-size: 2.75em;
 font-family: Georgia, "Times New Roman", Times, serif;
 font-weight: normal;
 text-align: center;
 letter-spacing: 1px;
 color: rgb(133,161,16);
 text-transform: uppercase;
 background-color: rgb(226,235,180);
}
```

If you preview the page now, you'll see that the headline has a light green background. When applied to a block-level element like a headline, the background fills the entire horizontal space available (in other words, the color doesn't just sit behind the text "The Amazing World of CSS," but extends all the way to the right edge of the box).

The headline text is a little cramped—the "T" that begins the headline touches the edge of the background. With a little padding, you can fix this.

3. **Add another property to the `h1` tag style so that it looks like this (changes in bold):**

```
h1 {
 font-size: 2.75em;
 font-family: Georgia, "Times New Roman", Times, serif;
 font-weight: normal;
 text-align: center;
 letter-spacing: 1px;
 color: rgb(133,161,16);
 text-transform: uppercase;
 background-color: rgb(226,235,180);
 padding: 5px 15px 2px 15px;
}
```

The `padding` shorthand property provides a concise way to add padding around all four sides of the content—in this case, 5 pixels of space are added above the text, 15 pixels to the right, 2 pixels to the bottom, and 15 pixels to the left.

There's one other problem with the headline: Because of the padding added to the <body> tag (see step 4 on page 218), the headline (including its background color) is indented 15 pixels from the left and right edges of the gray border surrounding the body. The headline would look better if its background color touched the gray border. No problem; negative margins to the rescue.

4. **Add one last property to the `h1` tag style so that it looks like this (changes in bold):**

```
h1 {
 font-size: 2.75em;
 font-family: Georgia, "Times New Roman", Times, serif;
 font-weight: normal;
 text-align: center;
 letter-spacing: 1px;
 color: rgb(133,161,16);
 text-transform: uppercase;
 background-color: rgb(226,235,180);
 padding: 5px 15px 2px 15px;
 margin: 0 -15px 20px -15px;
}
```

Here, the margin shorthand sets the top margin to 0, the right margin to –15 pixels, the bottom margin to 20 pixels, and the left margin to –15 pixels. The bottom margin just adds a bit of space between the headline and the paragraph that follows. The next trick is the use of negative values for the left and right margins. You can assign a negative margin to any element. This property pulls the element out toward the direction of the margin—in this case, the headline extends 15 pixels to the left and 15 pixels to the right, actually expanding the headline and pulling it out over the <body> tag's padding.

5. **Now, you'll add some formatting of the `<h2>` tags. Add the following style after the `h1` tag style:**

```
h2 {
 font-size: 1.5em;
 font-family: "Arial Narrow", Arial, Helvetica, sans-serif;
 color: rgb(249,107,24);
 border-top: 2px dotted rgb(141,165,22);
 border-bottom: 2px dotted rgb(141,165,22);
 padding-top: 5px;
 padding-bottom: 5px;
 margin: 15px 0 5px 0;
}
```

This style adds some basic text formatting and a dotted border above and below the headline. To add a bit of space between the headline text and the lines, it puts a small bit of padding at the top and bottom. Finally, the `margin` property adds 15 pixels above the headline and 5 pixels below it.

6. **Save the file and preview the page in a web browser.**

The headlines are looking good (see Figure 7-20). Next, you'll create a sidebar on the right side of the page.

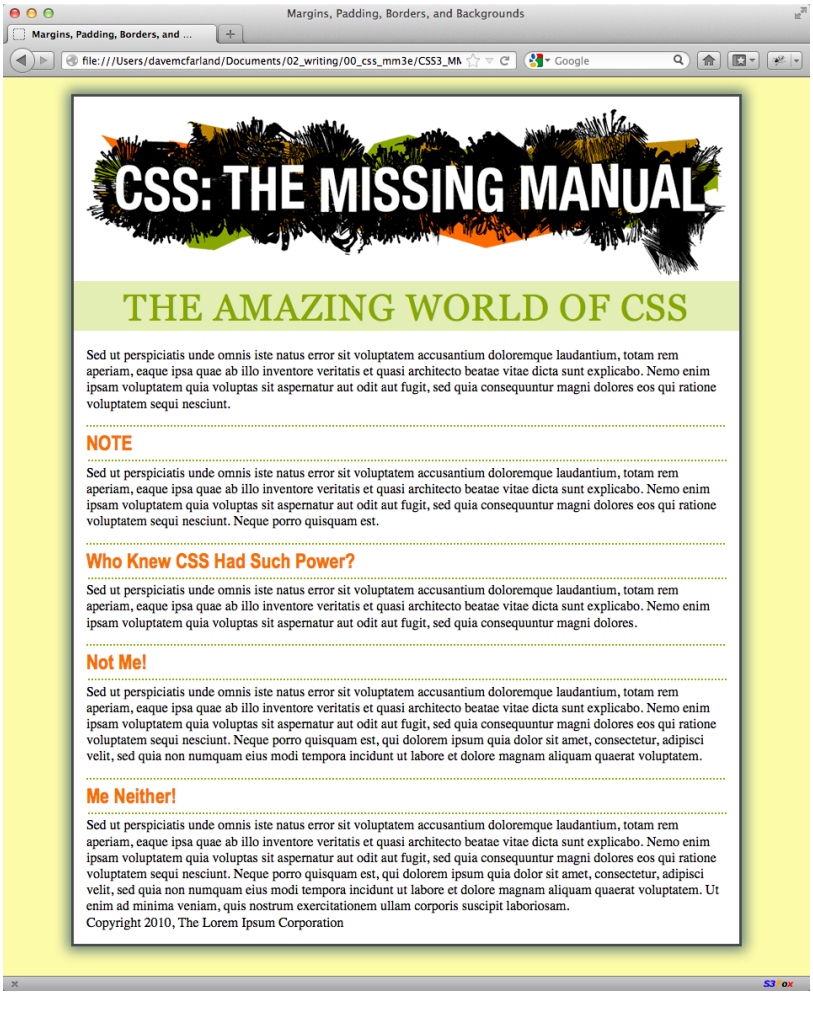

FIGURE 7-20
With just a few styles, you can add background colors, control margins throughout the page, and adjust the space between headlines and paragraphs.

Building a Sidebar

Sidebars are common elements in most types of print publications like magazines, books, and newspapers. They compartmentalize and highlight small chunks of information like a resource list, contact information, or a related anecdote. But to be effective, sidebars shouldn't interrupt the flow of the main story. They should, like the name says, sit unobtrusively off to one side, which you can easily make happen with CSS.

Instead of the div class sidebar, you should use the aside tag. Update your styles to reflect this change to this part of the code.

1. **Return to your text editor and open the *index.html* file.**

   First, you must isolate the region of the page that makes up the sidebar. The tag is the perfect tool. You can enclose any amount of HTML into its own self-contained chunk by wrapping it in a tag.

2. **Scroll down the page into the HTML and click *before* the first <h2> tag (the one with the "NOTE" headline). Then type , and press Enter (Return).**
   This HTML marks the beginning of the sidebar and applies a class to it. You'll create the .sidebar class style soon, but first you need to indicate the end of the sidebar by closing the .

3. **Click after the closing </p> tag that immediately follows the <h2> tag (this is the </p> that appears just before <h2>Who Knew CSS Had Such Power?</h2>). Press Enter, and then type .**
   You've just wrapped a headline and paragraph inside a tag. Next, you'll create a style for it.

4. **Return to the *main.css* file. Add the following style below the h2 style you created earlier:**

```
.sidebar {
 width: 30%;
 float: right;
 margin: 10px;
}
```

Use the aside tag instead of the class .sidebar

   This style sets the width of the content area (where the text appears) to 30 percent. In this case, the sidebar's width is 30 percent of the width of the container. The container is the <body> tag and its width will be up to 760 pixels (see step 4 on page 18.) The float property moves the sidebar to the right side of the box, and the margin property adds 10 pixels of space around the sidebar.

   If you preview the page in a browser, you'll see that the basic shape and placement of the sidebar are set, but there's one problem: The borders from the <h2> tags appear *underneath* the box. Even though the floated sidebar moves the text of the headlines out of the way, floats don't displace borders or backgrounds. Those just appear right under the floated sidebar. One way to fix this problem is to simply add a background color to the sidebar, so you can't see the h2 borders. (There's another technique, as well, which you'll use in step 9 on page 226.)

5. **Add two other properties to the .sidebar style so it looks like this (changes in bold):**

```
.sidebar {
 width: 30%;
 float: right;
 margin: 10px;
```

```
 background-color: rgb(250,235,199);
 padding: 10px 20px;
}
```

These properties add a light orangish color to the sidebar and indents the text from the sidebar's edges so it won't touch the borders you're about to add.

6. **Add two more properties to the `.sidebar` style so it looks like this (changes in bold):**

```
.sidebar {
 width: 30%;
 float: right;
 margin: 10px;
 background-color: rgb(250,235,199);
 padding: 10px 20px;
 border: 1px dotted rgb(252,101,18);
 border-top: 20px solid rgb(252,101,18);
}
```

Here's an example of the handy technique described on page 197. If you want most of the borders around an element to be the same, you can first define a border for all four edges—in this case a 1-pixel, dotted, orange line around the entire sidebar—and then supply new border properties for the specific edges you want changed—in this example, the top border will be 20 pixels tall and solid. This technique lets you use just two lines of CSS code instead of four (`border-top`, `border-bottom`, `border-left`, and `border-right`).

Next, you'll add rounded corners and a drop shadow to really make this sidebar stand out.

7. **Finally, add two more properties to the `.sidebar` style so it looks like this (changes in bold):**

```
.sidebar {
 width: 30%;
 float: right;
 margin: 10px;
 background-color: rgb(250,235,199);
 padding: 10px 20px;
 border: 1px dotted rgb(252,101,18);
 border-top: 20px solid rgb(252,101,18);
 border-radius: 10px;
 box-shadow: 5px 5px 10px rgba(0,0,0,.5);
}
```

The `border-radius` property (page 199) lets you create rounded corners. In this case, the 10-pixel setting provides a prominent-looking curve. The `box-shadow` property here adds a drop shadow below and to the right of the box, making it look as though it's floating above the page. You're almost done.

The headline inside the sidebar doesn't look quite right. It uses the same properties as the other <h2> tags (because of the h2 tag style you created in step 4). The border is distracting and the top margin pushes the headline down too much from the top of the sidebar. Fortunately, you can use a descendant selector to override those properties.

8. **After the `.sidebar` style, in the *main.css* style sheet, add a descendant selector:**

```
.sidebar h2 {
 border: none;
 margin-top: 0;
 padding: 0;
}
```

Because of the `.sidebar`, this style is more powerful—that is, it has greater *specificity* as described on page 102—than the basic h2 style. It erases the border from the original h2 tag style, along with the top margin and all the padding. However, since this style doesn't have a font size, color, or font family, those properties from the h2 style still apply—it's the cascade in action!

The page is looking good, but the borders on the <h2> tags still run up to and behind the sidebar. That just doesn't look good, but you can fix it easily.

9. **Locate the <h2> style and add the `overflow` property, like so:**

```
h2 {
 font-size: 1.5em;
 font-family: "Arial Narrow", Arial, Helvetica, sans-serif;
 color: rgb(249,107,24);
 border-top: 2px dotted rgb(141,165,22);
 border-bottom: 2px dotted rgb(141,165,22);
 padding-top: 5px;
 padding-bottom: 5px;
 margin: 15px 0 5px 0;
 overflow: hidden;
}
```

Setting the `overflow` property to `hidden` hides the borders that pass beyond the headline text and under the floating element.

10. **Save the file and preview the web page in a browser.**

The page should look like Figure 7-21.

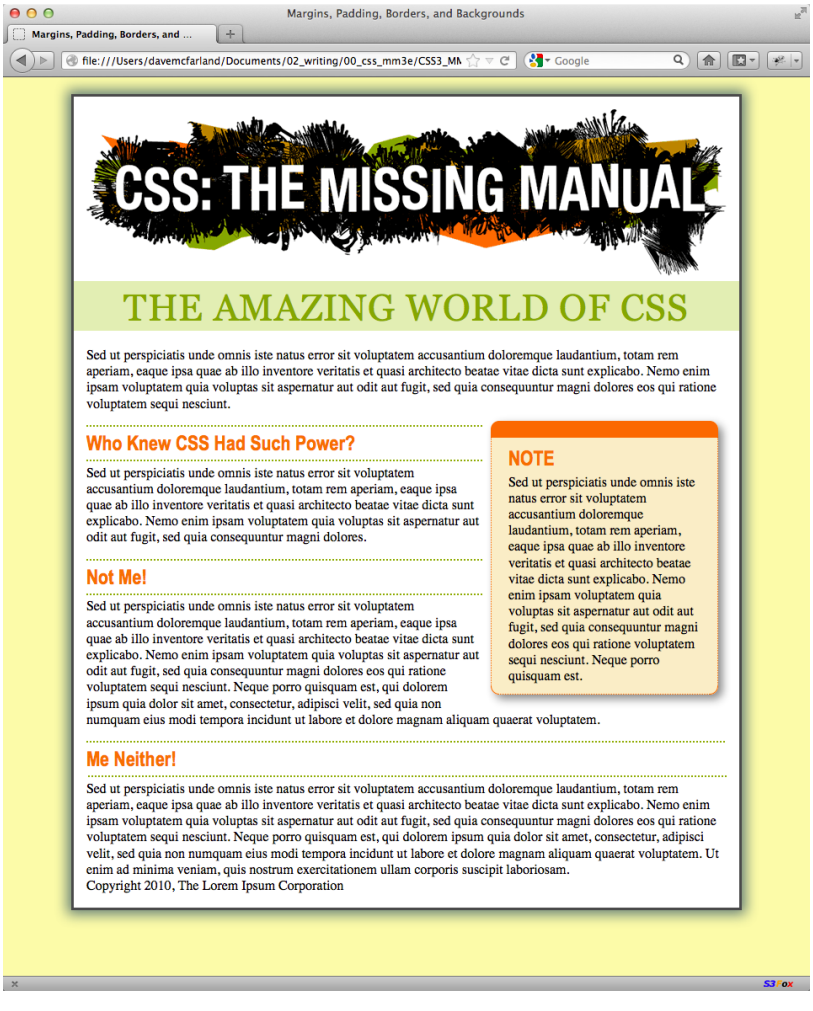

FIGURE 7-21
A handful of CSS styles add design elegance to ho-hum HTML. Notice how the floated sidebar both attracts attention and moves it out of the way of the main body of text.

Going Further

To try out your newfound skills, try this exercise on your own: Create a p tag style to add some pizzazz to the paragraphs on the page—try out some margin settings, font color, and so on. Next, create a class style for formatting the copyright notice that appears at the bottom of the *index.html* page (called, say, `.copyright`). In this style, add a border above the copyright notice, change its text color, shrink its font size, and change the type to uppercase. (Hint: Use the `text-transform` property discussed on page 157.) After you've created the style, add the appropriate class attribute to the <p> tag in the HTML.