

Quick Solutions to Common CSS Problems

**Covers
CSS 2.1**



CSS Cookbook

O'REILLY®

Christopher Schmitt

Page Elements

2.0 Introduction

From the most obvious design elements, such as the font and leading used in paragraphs and headings, to those that are often overlooked, such as the size of the margins, every element you place in the layout of a web page adds to the intended message of the content being displayed.

This chapter covers the page elements that comprise a web page. *Page elements* are items that affect the appearance of a web page, but aren't necessarily a part of the page. For example, a border around the viewport, the area of a web page that is seen by the user in the web browser, is a page element.

By manipulating elements such as the margins and borders surrounding a web page, you effectively frame the content of the page without actually styling the content. Such simple changes can affect the page's overall design in a profound way, or they can add that final, subtle detail that completes the design.

2.1 Eliminating Page Margins

Problem

You want to get rid of the whitespace around the edges of a web page and between the browser chrome and the contents of the page, as shown in Figure 2-1.

Solution

Set the value of the margin and padding properties for the `html` and `body` elements to 0:

```
html, body {  
  margin: 0;  
  padding: 0;  
  position: absolute;
```

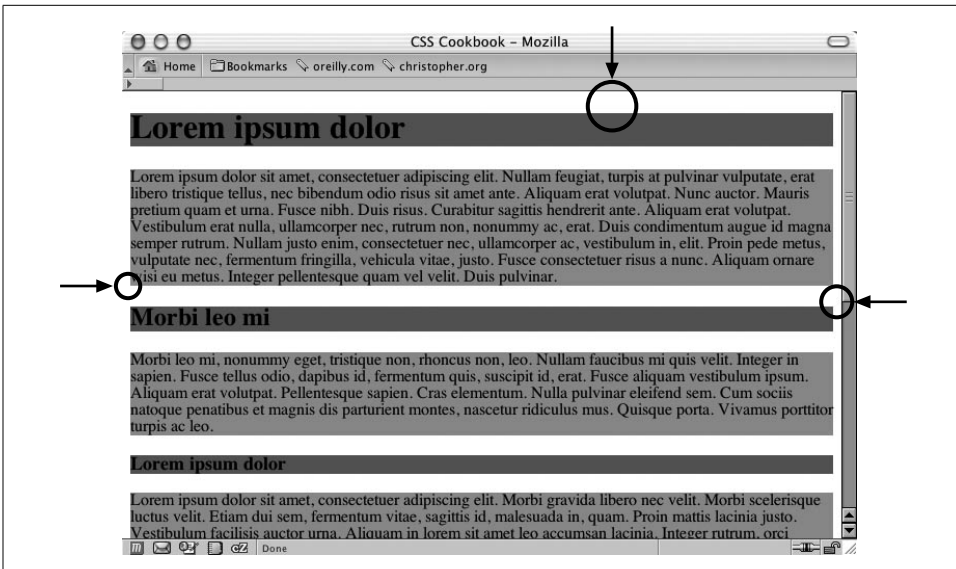


Figure 2-1. Page margins visible as the whitespace around the edges of a web page

```
top: 0;
left: 0;
}
```

Discussion

Setting the margin and padding properties of the body element to 0 helps create a full-bleed effect—in other words, it eliminates the whitespace around a web page (the units in this case don't matter). And setting the position to absolute and the values for top and left to 0 helps remove the body margins in Netscape Navigator 4.

However, depending on the content of the web page, the margin and padding properties might not be all you need to change to get a full-bleed effect. Default properties on other elements can have unexpected side effects when attempting to change the page margin. For example, if h1 is the body element's first child element, some unintended whitespace will appear above the headline and below the top of the browser's viewport. Figure 2-2 shows this undesired effect; the background color of the headings and paragraphs is gray so that you can more clearly see the effect.

To ensure the full-bleed effect in this situation you should set the margin and padding of the offending element (in this case, h1, h2, h3) to 0 as well as the body. This sets all the sides of the element's padding to 0. If that setup isn't possible (for example, you need to have a value at the bottom padding or margin), set the margin-top and padding-top values to 0 to maintain the full-bleed effect:

```
body {
margin: 0;
padding: 0;
```

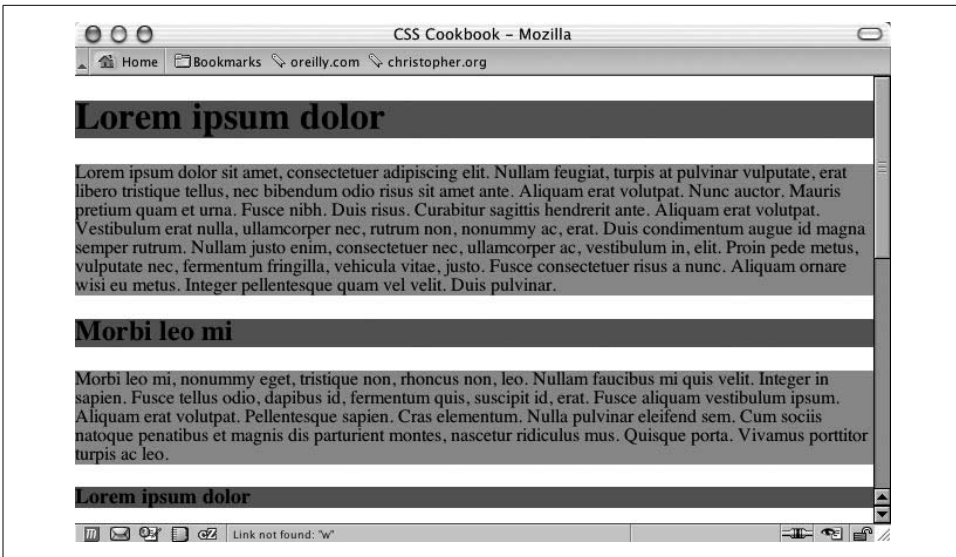


Figure 2-2. Whitespace above the heading and below the top of the browser's viewport

```
}  
h1, h2, h3 {  
  margin-top: 0;  
  padding-top: 0;  
  background-color: #666;  
}  
p {  
  background-color: #999;  
}
```

As you can see in Figure 2-3, this accomplishes the full-bleed effect. Notice how the gray background color of the first heading now touches the top of the browser's viewport.

See Also

Recipe 7.2 for writing one-column layouts by setting the margin and padding properties to a value other than 0.

2.2 Coloring the Scrollbar

Problem

You want to adjust the color of the scrollbar on a browser's viewport, or the window on the browser.

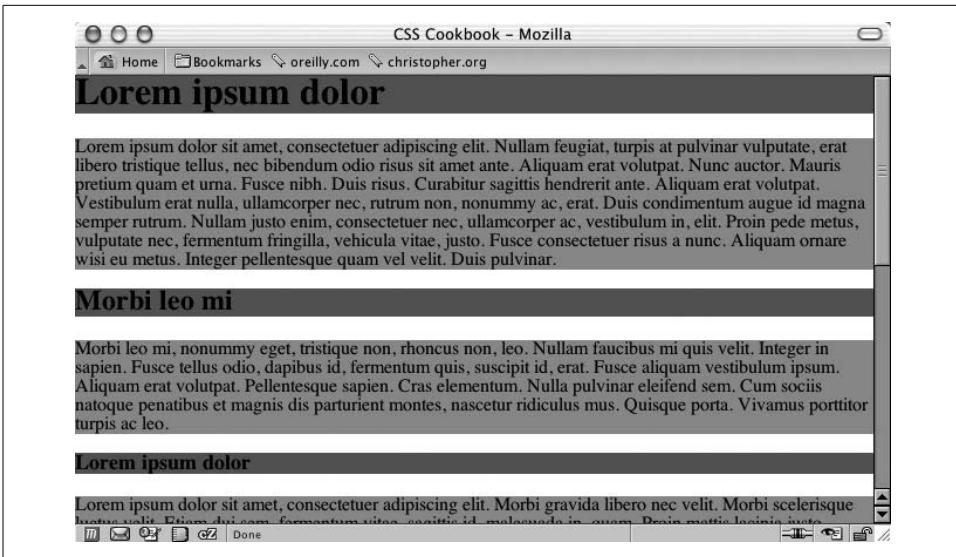
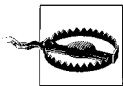


Figure 2-3. Whitespace removed above the heading

Solution

Use the properties that manipulate scrollbar colors in browsers that support it:

```
body,html {
  scrollbar-face-color: #99ccff;
  scrollbar-shadow-color: #ccccff;
  scrollbar-highlight-color: #ccccff;
  scrollbar-3dlight-color: #99ccff;
  scrollbar-darkshadow-color: #ccccff;
  scrollbar-track-color: #ccccff;
  scrollbar-arrow-color: #000033;
}
```



Because these properties aren't part of the W3C recommendations for CSS, browser vendors don't have to put in support for these properties. This Solution works only on the KDE Konqueror browser and on Internet Explorer 5.5+ for Windows. Other browsers will skip over the rules as though they weren't there. These rules won't be validated by services such as <http://jigsaw.w3.org/css-validator/validator-uri.html>.

Discussion

Although you might think of a scrollbar as a simple tool, it's actually composed of several widgets that create a controllable 3D object. Figure 2-4 spotlights the different properties of a scrollbar. As you can see, to create a truly different color scheme for the scrollbar, you must alter the value of seven properties.

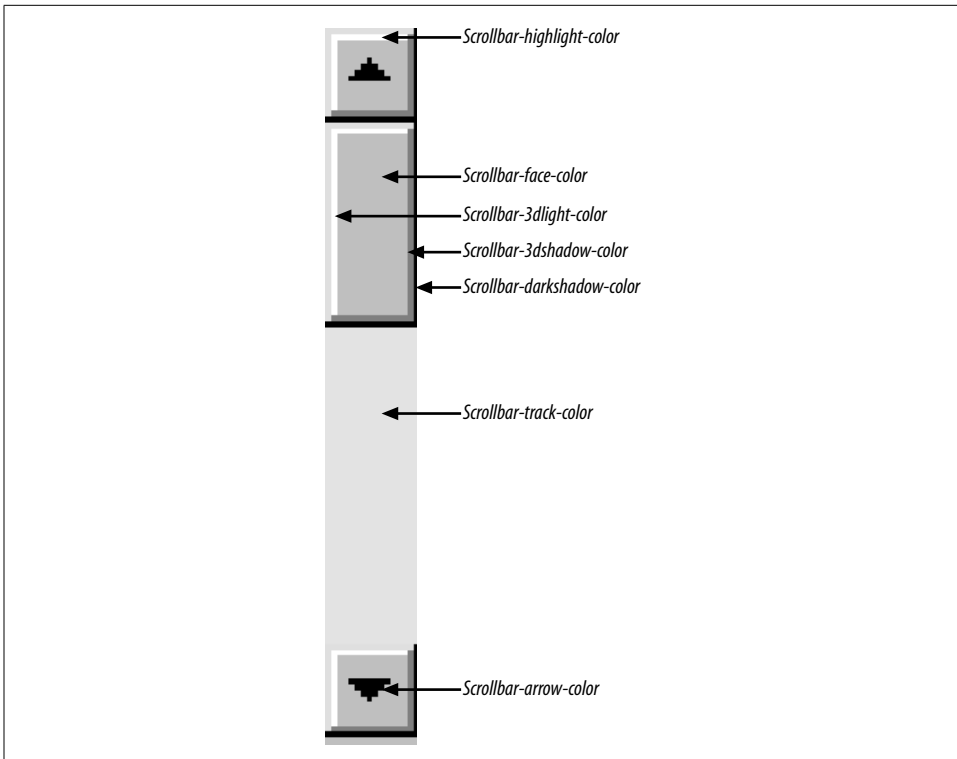


Figure 2-4. The parts of a scrollbar that can be affected by proprietary CSS for Internet Explorer for Windows

In addition to adjusting the scrollbar of the browser viewport, you also can adjust the colors of the scrollbar in the textarea for a web form, framesets, iframes, and generally anything with a scrollbar:

```
.highlight {
  scrollbar-face-color: #99ccff;
  scrollbar-shadow-color: #ccccff;
  scrollbar-highlight-color: #ccccff;
  scrollbar-3dlight-color: #99ccff;
  scrollbar-darkshadow-color: #ccccff;
  scrollbar-track-color: #ccccff;
  scrollbar-arrow-color: #000033;
}

<form>
  <textarea class="highlight"></textarea>
</form>
```

When rendering a page that doesn't contain a valid DOCTYPE, Internet Explorer for Windows experiences what is known as *quirks* (nonstandard behavior) mode and looks for the scrollbar properties in the body selector. When the page contains a valid

DOCTYPE, Internet Explorer for Windows is in Standards mode and it obeys the `html` selector. So, just in case the web document's DOCTYPE might change, it's best to ensure that the `body` and `html` selectors are grouped and applied in one CSS rule:

```
html .highlight, body .highlight {
  scrollbar-face-color: #99ccff;
  scrollbar-shadow-color: #ccccff;
  scrollbar-highlight-color: #ccccff;
  scrollbar-3dlight-color: #99ccff;
  scrollbar-darkshadow-color: #ccccff;
  scrollbar-track-color: #ccccff;
  scrollbar-arrow-color: #000033;
}
```

See Also

The MSDN Scrollbar Color Workshop at <http://msdn.microsoft.com/workshop/samples/author/dhtml/refs/scrollbarColor.htm> to pick colors for a custom scrollbar; Recipe 3.3 for changing the cursor, another user interface widget of the browser.

2.3 Centering Elements on a Web Page

Problem

You want to center parts of a web page, as in Figure 2-5.



Figure 2-5. The headline text centered

Solution

To center text in a block-level element, use the `text-align` property:

```
h1, h2, h3 {
    text-align:center;
}
```

Discussion

By using `text-align`, you can center text inside block-level elements. However, in this example, the heading takes up the entire width of the body element, and if you don't apply a background color to the element, you probably won't even notice this is happening. The gray background color in Figure 2-6 shows the actual width of the centered elements.

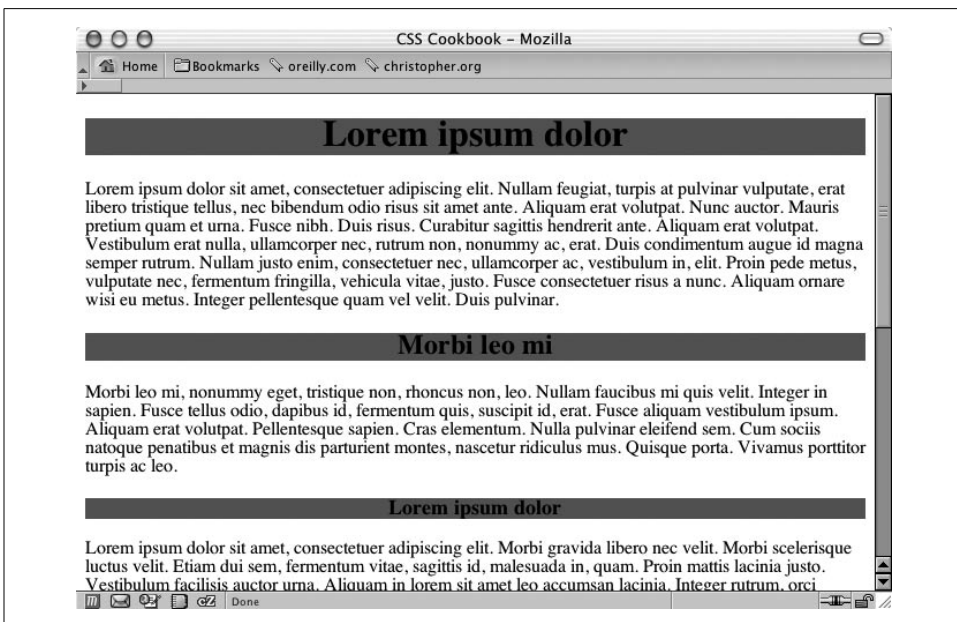


Figure 2-6. The actual width of the elements shown by the gray background color

An alternative approach is to use margins to center text within its container:

```
h1, h2, h3 {
    margin-left: auto;
    margin-right: auto;
}
```

When you set the `margin-left` and `margin-right` properties to `auto`, you center the element inside its parent element. However, older but still popular browsers won't render the presentation correctly. So, workarounds are needed for individual situations.

Tables

To center a table, place the table as the child of a div element:

```
<div class="center">
  <table width="50%" border="1" cellpadding="30">
    <tr>
      <td>This is the first cell</td>
      <td>This is the second cell</td>
    </tr>
    <tr>
      <td>This is the third cell, it's under the first cell</td>
      <td>This is the fourth cell, it's under the second cell.</td>
    </tr>
  </table>
</div>
```

Then write the following CSS rule:

```
.center {
  text-align: center;
}
.center table {
  width: 50%;
  margin-left: auto;
  margin-right: auto;
  text-align: left;
}
```

Although setting both sides of the margin to auto works in newer generations of browsers, it doesn't work in Internet Explorer 5 for Windows or Netscape Navigator 4. To catch those two browsers and tell them to "do the right thing," the center class selector uses the text-align technique. However, if that were all you did, the contents of the table cells would be centered as well. To counteract that effect, use a descendent selector, `.center table`, to align the contents in the table cell elements.

Note that if you use th elements in an HTML table, the content inside those cells is centered by default. Setting the text-align property to a value of left in the descendent selector `.center table` doesn't counter that effect. To left-align the content inside th, use this CSS rule:

```
th {
  text-align: left;
}
```

To save a line or two of CSS code, you might want to incorporate the shorthand version of the margin property, as shown here (although this works in most browsers, it fails in Internet Explorer 5 for Macintosh):

```
.center table {
  margin: 0 auto;
  text-align: left;
}
```

Images

If you want to center an image, wrap a `div` element around the `img` element first. This technique is required because an `img` element, like `em` and `strong`, is inline. It rests in the flow of the web page instead of marking off space like the `p` or `blockquote` block-level elements do. The markup looks like this:

```
<div class="flagicon"></div>
```

And the CSS rule looks like this:

```
.flagicon {
  text-align: center;
}
```

To center elements with fixed widths, such as images, first set the value of the parent's `padding-left` property to 50%. Then determine half of the width of the element you are centering and set it as a negative value in the `margin-left` property. That prevents the element's left side from resting on the 50% line caused by its padding and makes it slide into the middle of the page. The markup for an image in a web page using this technique looks something like this:

```

```

The CSS rule to produce the result shown in Figure 2-7 looks like this:

```
body {
  padding-left: 50%;
}
img {
  /* equal to the negative of half its width */
  margin-left: -138px;
}
```



Figure 2-7. The image centered without the deprecated `center` element

Vertical centering

With the element centered horizontally, you can take this technique one step further and center the image (or any other element) vertically as well. The difference with this method is that it uses the `position` property to make this work. The markup is the same as that used for the image element in the previous example, but this time the CSS rule is for just one selector (see Figure 2-8):

```
img {
  position: absolute;
  top: 50%;
  left: 50%;
  margin-top: -96px;
  margin-left: -138px;
  height: 192px;
  width: 256px;
}
```



Figure 2-8. The image centered horizontally and vertically on the web page

With absolute positioning, you take the element out of the normal flow of the document and place it wherever you want.

If you want to center both text and an image (or other images) instead of just one image, enclose all the content with a `div` element:

```
<div id="centerFrame">
  <p>Epsum factorial non deposit quid pro quo hic escorol. Olypian
  quarrels et gorilla congolium sic ad nauseum. Souvlaki ignitus
  carborundum e pluribus unum. Defacto lingo est igpay atinlay.</p>
  
</div>
```

Then in the CSS rule, remove the height property and adjust the negative value of the top margin to compensate for the additional elements on the page:

```
#centerFrame {
  position: absolute;
  top: 50%;
  left: 50%;
  /* adjust negative value until content is centered */
  margin-top: -150px;
  margin-left: -138px;
  width: 256px;
}
```

Keep the amount of content that you want centered short. If you have numerous images and long amounts of HTML text, users with small resolutions will have to scroll the page to see your centered content.

See Also

Chapter 7 for information on multicolumn layouts, which deal with the position of elements in a web page; the CSS 2.1 specification for text-align at <http://www.w3.org/TR/CSS21/text.html#propdef-text-align>.

2.4 Setting a Background Image

Problem

You want a background image that doesn't repeat.

Solution

Use the background-image and background-repeat properties to control the display of an image (see Figure 2-9):

```
body {
  background-image: url(bkgd.jpg);
  background-repeat: no-repeat;
}
```

Discussion

You can place text and other inline images over a background image to create a sense of depth on a web page. Also, you can provide a framing device for the web page by tiling a background image along the sides of a web browser.

See Also

Recipe 2.5 for repeating background images in a line either horizontally or vertically.

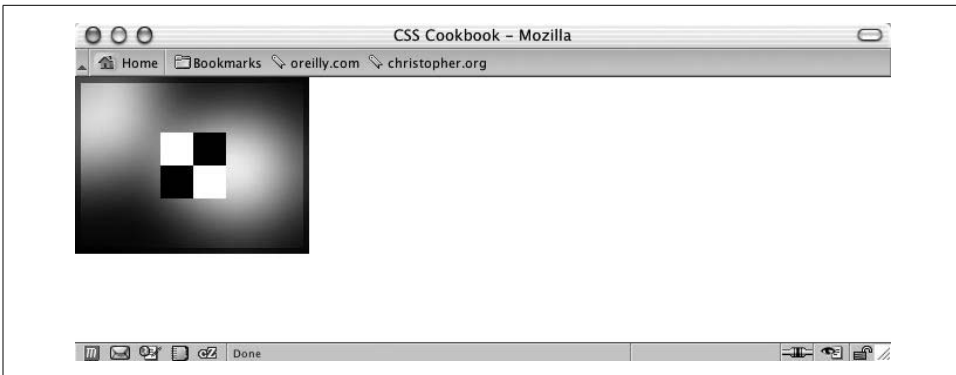


Figure 2-9. The background image displayed once in the upper right corner

2.5 Creating a Line of Background Images

Problem

You want a series of background images to repeat vertically or horizontally.

Solution

To tile the background image horizontally, or along the x axis, use the following CSS rule (see Figure 2-10):

```
body {  
  background-image: url(bkgd.jpg);  
  background-repeat: repeat-x;  
}
```

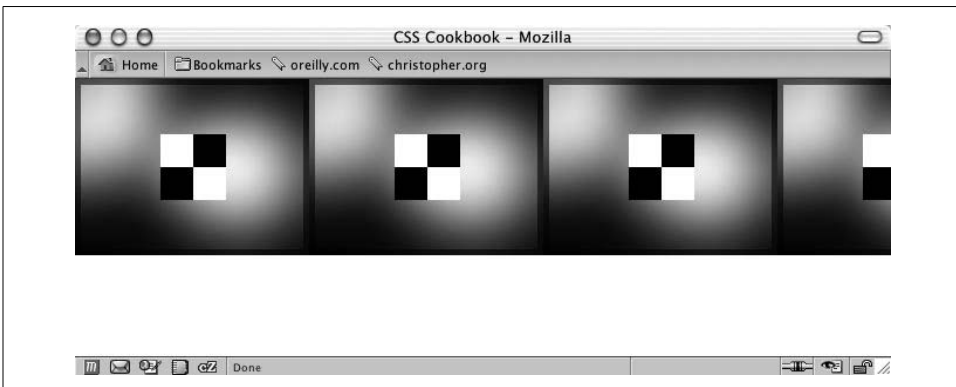


Figure 2-10. The background image tiled horizontally

To have the background image repeat along the vertical axis, use the `repeat-y` value for `background-repeat`.

See Also

Recipe 2.6 for placing a background image at a specific location in a web page.

2.6 Placing a Background Image

Problem

You want to position a background image in a web page.

Solution

Use the `background-position` property to set the location of the background image. To place an image that starts 75 pixels to the right and 150 pixels below the upper-left corner of the viewport (see Figure 2-11), use the following CSS rule:

```
body {  
    background-image: url(bkgd.jpg);  
    background-repeat: no-repeat;  
    background-position: 75px 150px;  
}
```

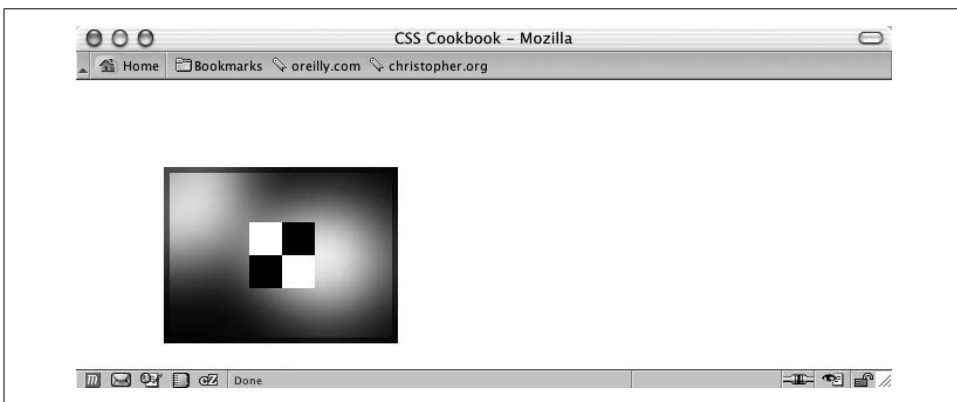


Figure 2-11. The background placed precisely 75 pixels from the right and 150 pixels from the upper left corner of browser's viewport

Discussion

The `background-position` element contains two values separated by a space. The first value of the pair sets the origin point along the y axis, while the second value sets the point on the x axis. If only one value is given, that value is used for the horizontal position and the vertical position is set to 50%.

The Solution used pixel units to determine the placement of the background image; however, you also can use percentages. A value of 50% for `background-position` means that the browser places the image in the dead center of the viewport, as

shown in Figure 2-12, while the values 0% and 100% place the image in the upper left and lower right corner, respectively.



Figure 2-12. The background image centered in the browser window

Along with percentages, you can use the values `top`, `center`, and `bottom` for the y axis and `left`, `center`, and `right` for the x axis. Using combinations of these values, you can place the background image at the eight points around the edges of the viewport (in the corners and in between), as well as in the middle of the viewport. For example, to re-create the value of 50% in Figure 2-12, you can use this CSS rule instead:

```
body {
  background-image: url(bkgd.jpg);
  background-repeat: no-repeat;
  background-position: center center;
}
```

To place a background image in the lower right corner, as shown in Figure 2-13, you can use the following CSS rule:

```
body {
  background-image: url(bkgd.jpg);
  background-repeat: no-repeat;
  background-position: bottom right;
}
```

You also can use the `background-position` and `background-repeat` properties for background images that tile but aren't chained to the sides of the viewport. For example, the following CSS snippet creates a web page design such as that shown in Figure 2-14:

```
body {
  background-image: url(montage.jpg);
  background-repeat: repeat-x;
  background-position: 55px 100px;
}
h1 {
  font-size: 75px;
  font-family: Verdana, Helvetica, Arial, sans-serif;
```

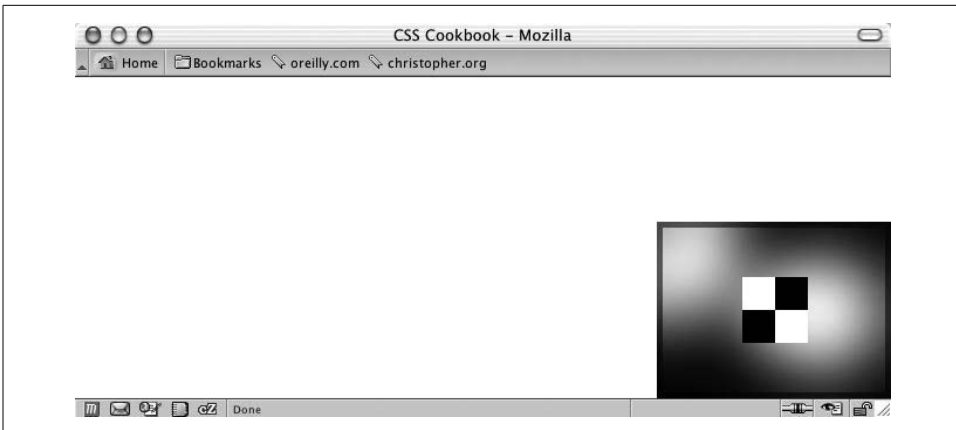


Figure 2-13. The background image placed in the lower right corner

```
text-align: center;
margin: 0;
padding: 0 0 125px 0;
}
p {
line-height: 1.5em;
font-family: Verdana, Helvetica, Arial, sans-serif;
margin: 0 15%;
}
```



Figure 2-14. A repeating montage created using the CSS properties `background-repeat` and `background-position`

Note that Netscape Navigator 4 doesn't support `background-position`, and it's impossible to work around this limitation through CSS.

See Also

Recipe 2.7 for setting an image so that it doesn't scroll; the CSS 2.1 specification for `background-position` at <http://www.w3.org/TR/CSS21/colors.html#propdef-background-position>.

2.7 Fixing the Background Image

Problem

You want a background image to remain in the browser window, even as the user scrolls down a web page.

Solution

Use the `background-attachment` property set with a fixed value, like so:

```
body {
  background-image: url(bkgd.jpg);
  background-repeat: no-repeat;
  background-attachment: fixed;
}
```

Discussion

By using this technique, you are locking down the background image. So, even if a visitor scrolls, the image remains where you placed it originally. Another acceptable value for `background-attachment` is `scroll`, which is the default value. So, even if you don't specify `scroll`, the background image moves up with the rest of the document as the visitor scrolls down.

For example, imagine you want to post on your web page a photo of a recent trip, and you want the photo positioned on the left side of the page and your text on the right. As the reader scrolls down to read more about the trip, the photo from the trip stays in place, as shown in Figure 2-15. Here's the code:

```
body {
  background-image: url(bkgd2.jpg);
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-position: -125px 75px;
  margin: 75px 75px 0 375px;
}
h1, h2, h3 {
  padding-top: 0;
  margin-top: 0;
  text-transform: uppercase;
}
```

```

}
p {
  text-align: justify;
}

```



Figure 2-15. The photo staying in place as the visitor scrolls

To take this further, you can lock down the image on block-level elements other than body. For example, try the heading elements when designing a review for a movie or concert. The following CSS rule can create the interesting surfing experience:

```

h1, h2, h3 {
  font-size: 200%;
  background-image: url(bkgd2.jpg);
}

```

```

background-repeat: no-repeat;
background-attachment: fixed;
background-position: center;
padding: 1.5em;
text-align: center;
color: white;
}

```

Because of the padding and light color on the headings, there is enough room to see the background image “through” the elements as well as to read the headlines. As the visitor scrolls the web page reading the review, she will see the rest of the image, as shown in Figure 2-16.

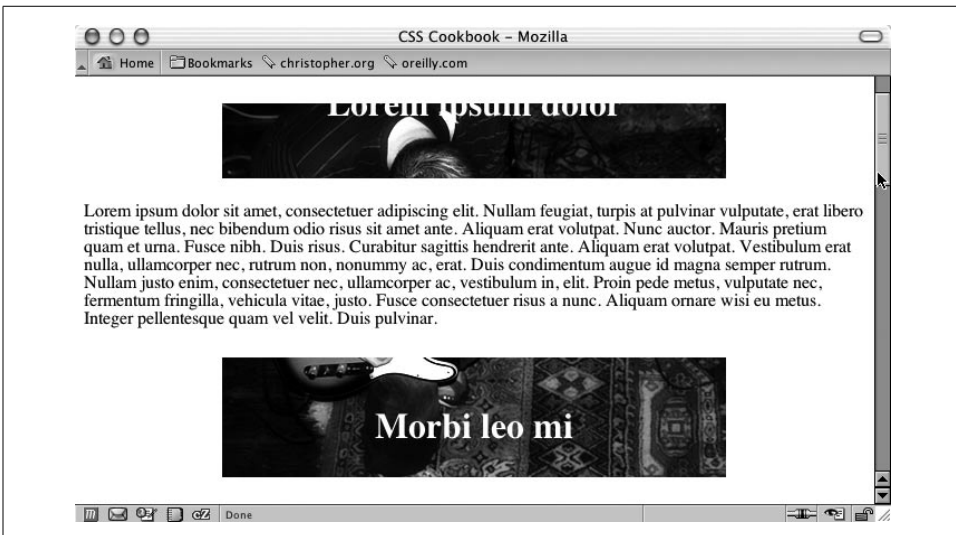


Figure 2-16. The photo coming through the headings instead of the body element

At press time, only Mozilla and Netscape 6+ supported the application of background images as fixed attachments to block-level elements like header elements used in this Solution. Internet Explorer 5.x and 6 for Windows repeat the background image in each header element.

See Also

Recipe 2.6 to position a background image; Recipe 10.5 for a hack to fix Internet Explorer for Windows' lack of support for background-fixed; the CSS 2.1 specification for background-attachment at <http://www.w3.org/TR/CSS21/colors.html#propdef-background-attachment>.

2.8 Placing a Page Border

Problem

You want to place a visual frame or border around a web page, as in Figure 2-17.



Figure 2-17. A framed web page

Solution

Use the border property on the body element:

```
body {  
  margin: 0;  
  padding: 1.5em;  
  border: 50px #666 ridge;  
}
```

Discussion

The border property is a shorthand property, in that it enables you to set the width, color, and style of the border around an element in one step instead of three. If you didn't use this shorthand property in the preceding Solution, you would have to replace the line that reads `border: 50px #666 ridge;` with the following three lines:

```
border-width: 50px;  
border-color: #666;  
border-style: ridge;
```

You can create a framing effect with other styles as well, such as dotted, dashed, solid, double, groove, inset, and outset (see Figure 2-18).

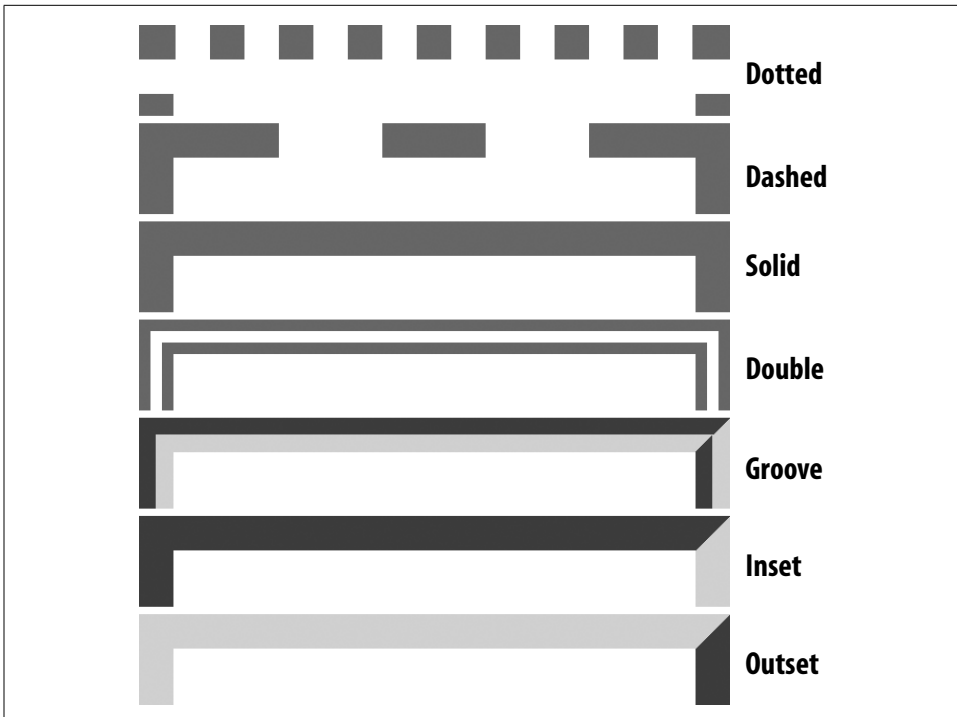


Figure 2-18. The available border styles in CSS

Note that groove style is the inverse of the shades of shadow as seen in the Solution, which uses the `ridge` value.

The only browser incompatibilities to worry about are that in Internet Explorer 5 for Macintosh and Internet Explorer for Windows, the dotted style appears as aliased circles, whereas in Netscape 6+, Mozilla, and Safari, the dotted style appears as blocks.

You also can place a stylized border on images as well. Instead of having a default solid line, try experimenting in your designs with groove or double borders as shown in Figure 2-19:

```
img.left {  
  float: left;  
  margin-right: 7px;  
  margin-bottom: 3px;  
  border: 4px double #666;  
}
```

See Also

Recipe 1.11 for creating pull quotes with different border styles.



Figure 2-19. A double border around an image

2.9 Customizing a Horizontal Rule

Problems

You want to change the look of a horizontal rule from the solid line in Figure 2-20 to something more interesting, for example the small centered rectangle in Figure 2-21.

Solution

Use a mixture of CSS properties on the `hr` element to obtain a desired effect:

```
hr {
  margin-left: auto;
  margin-right: auto;
  margin-top: 1.25em;
  margin-bottom: 1.25em;
  width: 10px;
  height: 10px;
  background-color: #777;
}
```

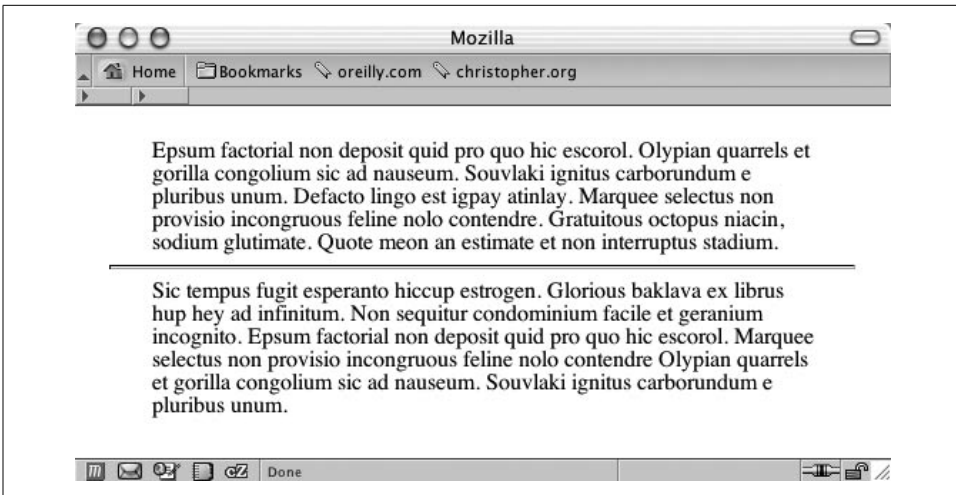


Figure 2-20. The default rendering of a horizontal rule

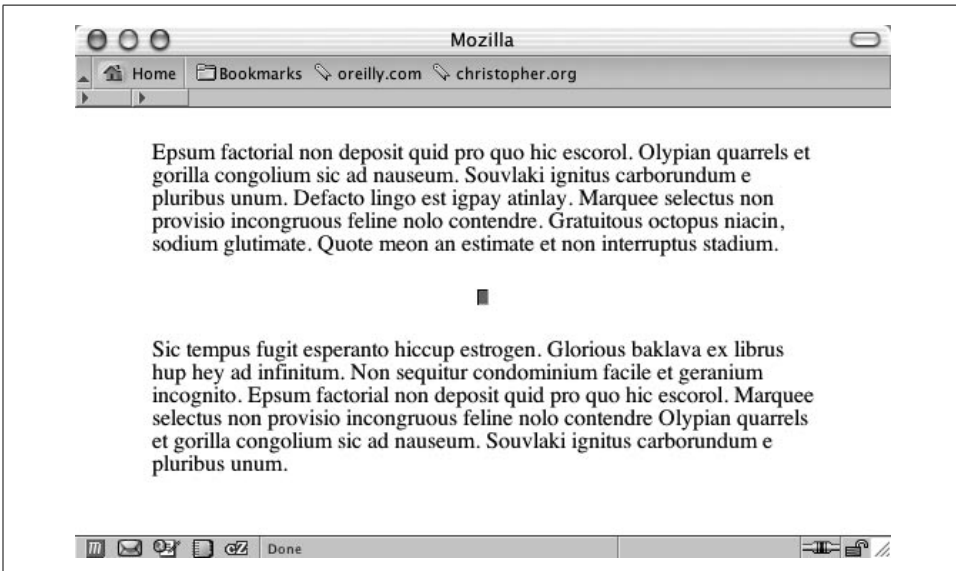


Figure 2-21. A stylized horizontal rule

Discussion

Before HTML 4.0, the presentation of horizontal rules could be manipulated through a set of four attributes: `align`, `width`, `size`, and `noshade`. Since HTML is intended to mark up content and not the look of the content, those values are no longer a part of the HTML specification. (Browser vendors may support the values, but your mileage will vary.) With CSS rules controlling the presentation, you have far greater control over the appearance of horizontal rules.

For example, you can set the height as well as the width properties for horizontal rules through CSS:

```
hr {
  width: 80%;
  height: 3px;
  margin-left: auto;
  margin-right: auto;
}
```

Setting the `margin-left` and `margin-right` to `auto` centers the horizontal rule in the web page for Safari, while it's not required for Mozilla, Navigator and Internet Explorer for Windows.

If you want to style an `hr` element with color (as shown in Figure 2-22), use the following code:

```
hr {
  color: green;
  background-color: green;
  width: 80%;
  height: 3px;
  margin-left: auto;
  margin-right: auto;
}
```

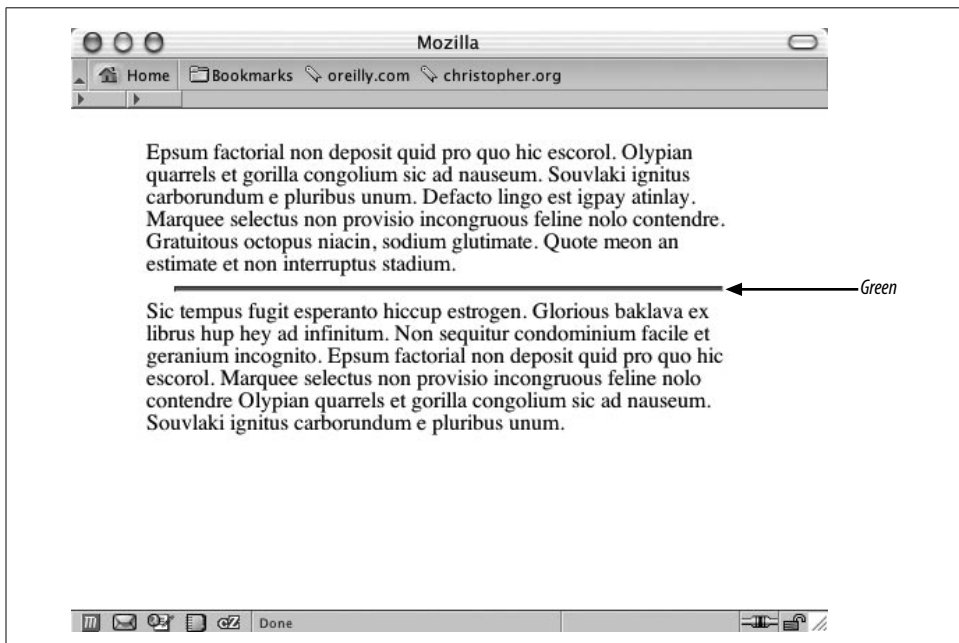


Figure 2-22. A centered, green horizontal rule

The first property, color, is understood by Internet Explorer for Windows while Safari, Mozilla, and Netscape Navigator 6+ pick up the second property, background-color.

To place an image instead of a horizontal bar, use the background-image property:

```
hr {  
  background-image: url(hr-decoration.gif);  
  background-repeat: no-repeat;  
  border: none;  
  width: 76px;  
  height: 25px;  
  margin-left: auto;  
  margin-right: auto;  
}
```

However, Internet Explorer for Windows renders a border around the hr element as shown in Figure 2-23 that can't be removed through CSS properties.

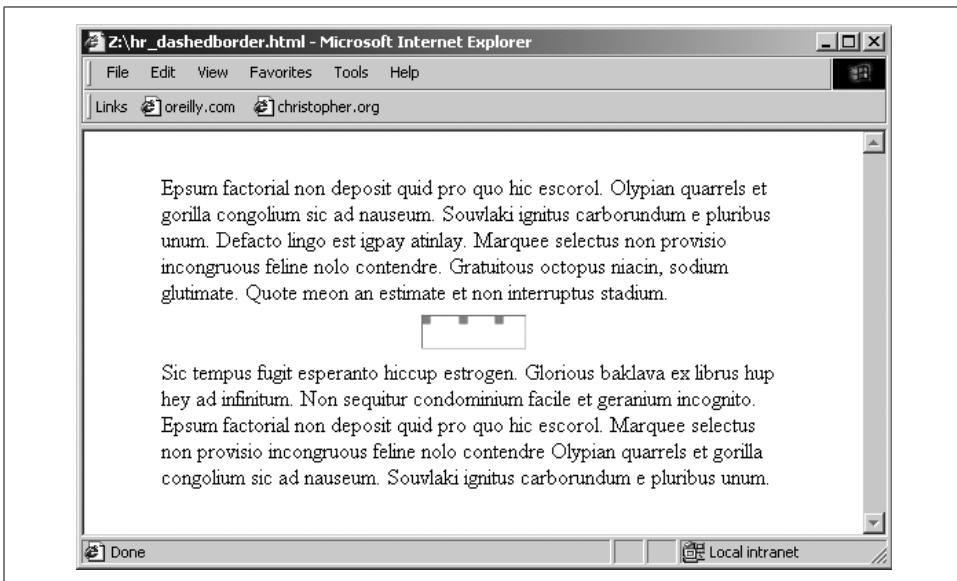


Figure 2-23. A border around a horizontal rule in Internet Explorer for Windows

See Also

The HTML 4.01 specification for hr elements at <http://www.w3.org/TR/html401/present/graphics.html#edef-HR>; an overview of styling an hr element at <http://www.sovavsi.cz/css/hr.html>; another example of refining the presentation of horizontal rules at http://www.sideshow.com/weblog/2004/03/17/sexily_styling_horizontal_rules.html.

2.10 Example Design: Setting Up a Dynamic Splash Page

Best suited for entertainment-related and personal web sites, a *splash page* is a web page typically comprising only an eye-catching image that is designed to entice visitors to enter a web site. Sometimes, however, that additional HTML page actually acts as a barrier to the content of the site. This example design remedies that problem.

The splash page in this example uses HTML elements from the existing main page of the web site. No separate HTML file is involved, so it appears as though there are two pages when there is only one. And with cookie detection built into the script, after a visitor sees the splash page once he won't see it again for at least another month (unless he deletes the cookie or tells his browser not to accept cookies).

Another benefit of the code in this section is that if the visitor's user agent doesn't handle JavaScript or JavaScript has been turned off manually, the visitor sees the default page design instead. He won't get trapped viewing only the splash page design, thereby locking him out from seeing your premium content on the main page.

Main Page

The first step is to create the design for the main page of your web site. Figure 2-24 shows an example. The code for *mainPage.css* is shown in Example 2-1.

Example 2-1. mainPage.css

```
body {
  margin: 0;
  background-color: white;
  padding-left: 0;
  padding-top: 0;
}
#logo {
  padding: 5% 20% 0.5em 5%;
  position: static;
  margin: 0;
}
#header h1 {
  margin: 0;
  padding: 0 0 0 5%;
  border-bottom: 1px solid black;
  font-family: Arial, Verdana, Helvetica, sans-serif;
}
#header h2 {
  margin: 0;
  padding: 0.5em 5% 0.5em 5%;
  font-size: 1em;
  text-align: right;
  border-bottom: 1px solid black;
  background-color: #ccc;
```

Example 2-1. *mainPage.css* (continued)

```
font-family: Arial, Verdana, Helvetica, sans-serif;
}
#header {
display: block;
}
#content {
margin: 0 5% 10% 5%;
font-size: 1.1em;
line-height: 1.6em;
display: block;
}
#footer {
border-top: 1px black solid;
padding: 1em;
text-align: center;
display: block;
}
```



Figure 2-24. *The main page design*

Having the default style sheet in place tells you which elements need to be addressed in the splash screen style sheet. Because you will be switching between the splash page and the main page style sheets, any selectors and their respective properties that appear in both must have their own respective values. Otherwise, a padding-left value of 50% for the body element dictated by the splash screen style sheet will carry over to the main page style sheet, moving all the content of the main page into the right half of the viewport.

The Splash Screen

The next step is to create a splash screen based on the marked-up content of the main page. To simplify things, you can copy the main page and link a new, blank *splashPage.css* style sheet. Then design a splash page based on the existing contents of the real page. In the example shown in Figure 2-25, the logo was carried over from the main page to the splash page and the rest of the page's content was hidden. The *splashPage.css* code, as shown in Example 2-2, creates this effect.

Example 2-2. *splashPage.css*

```
body {
  padding-left: 50%;
  padding-top: 15%;
}
#logo {
  margin-left: -73px;
}
#header, #content, #footer {
  display: none;
}
```

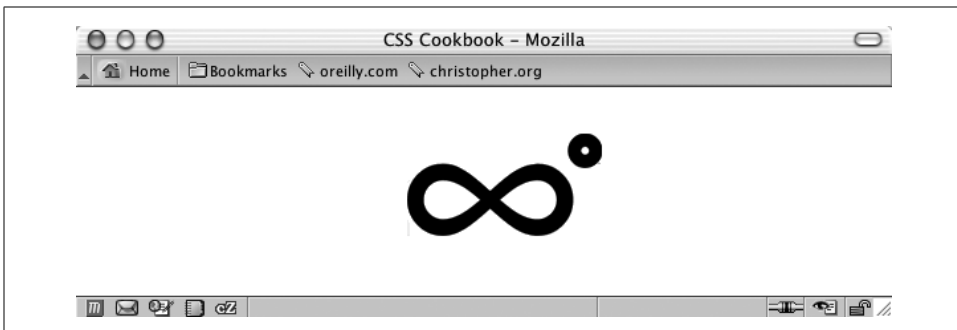


Figure 2-25. The splash page

Next, link these two separate style sheets to the main page HTML file. Use the `link` element to associate the default style sheet, *mainPage.css*, and then the style sheet that defines the design of the splash page, *splashPage.css*:

```
<link rel="stylesheet" type="text/css" media="all"
      href="mainPage.css" />
<link rel="alternate stylesheet" type="text/css" media="all"
      href="splashPage.css" title="splash" />
```

Switcher JavaScript

Now, add the alternative style sheet switcher JavaScript to your web page through the `src` attribute:

```
<script type="text/javascript" language="JavaScript"
      src="switcher.js"></script>
```

Example 2-3 shows the actual style sheet switcher code used in the *switcher.js*, which comes from Paul Swoden's Alternative Style Sheet Switcher at <http://www.alistapart.com/stories/alternate/>. How it works is beyond the scope of this book, but for more information on JavaScript, see *JavaScript: The Definitive Guide* (O'Reilly).

Example 2-3. *switcher.js*

```
function setActiveStyleSheet(title) {
    var i, a, main;
    for(i=0; (a = document.getElementsByTagName("link")[i]); i++) {
        if(a.getAttribute("rel").indexOf("style") != -1 && a.getAttribute("title")) {
            a.disabled = true;
            if(a.getAttribute("title") == title) a.disabled = false;
        }
    }
}

function getActiveStyleSheet() {
    var i, a;
    for(i=0; (a = document.getElementsByTagName("link")[i]); i++) {
        if(a.getAttribute("rel").indexOf("style") != -1 && a.getAttribute("title") && !a.disabled) return a.getAttribute("title");
    }
    return null;
}

function getPreferredStyleSheet() {
    var i, a;
    for(i=0; (a = document.getElementsByTagName("link")[i]); i++) {
        if(a.getAttribute("rel").indexOf("style") != -1
            && a.getAttribute("rel").indexOf("alt") == -1
            && a.getAttribute("title")
        ) return a.getAttribute("title");
    }
    return null;
}

function createCookie(name,value,days) {
    if (days) {
        var date = new Date();
        date.setTime(date.getTime()+(days*24*60*60*1000));
        var expires = "; expires="+date.toGMTString();
    }
    else expires = "";
    document.cookie = name+"="+value+expires+"; path=/";
}

function readCookie(name) {
    var nameEQ = name + "=";
    var ca = document.cookie.split(';');
    for(var i=0;i < ca.length;i++) {
        var c = ca[i];
        while (c.charAt(0)==' ') c = c.substring(1,c.length);
        if (c.indexOf(nameEQ) == 0) return c.substring(nameEQ.length,c.length);
    }
}
```

Example 2-3. switcher.js (continued)

```
}
return null;
}

window.onload = function(e) {
  var cookie = readCookie("style");
  var title = cookie ? cookie : getPreferredStyleSheet();
  setActiveStyleSheet(title);
}

window.onunload = function(e) {
  var title = getActiveStyleSheet();
  createCookie("style", title, 365);
}

var cookie = readCookie("style");
var title = cookie ? cookie : getPreferredStyleSheet();
setActiveStyleSheet(title);
```

Add an additional piece of JavaScript, as shown in Example 2-4, that loads the splash design over the default style.

Example 2-4. toggleSplash()

```
<script type="text/javascript" language="JavaScript">
function toggleSplash() {
  if (readCookie("splashCookie") == null) {
    setActiveStyleSheet('splash');
    createCookie("splashCookie", "noSplash", 31);
    timer=setTimeout("setActiveStyleSheet('default')",7000);
  }
}
</script>
```

To initiate the splash page when the visitor loads the web page, place an event trigger in the body element:

```
<body onload="toggleSplash();">
```

Compatibility

This splash screen works in Internet Explorer 5.5+ for Windows, Mozilla, Netscape 6+, and Internet Explorer 5 for Macintosh. The cookie detection method doesn't work in Safari, causing visitors to view the splash page each time the page is loaded.

Another way to approach splash page design is to include more elements from the main page. If you want to change the splash page design for your company web site—so that it shows a holiday message, for example—you can easily use heading elements that contain the name of the company and the tagline and then color them in orange and black text. Because the splash page design is in a separate style sheet, it's easy to modify and upload at any time.