You can identify specific file types in much the same way as external links or email links. But instead of looking for specific information at the beginning of the link's URL, you can find it at the end. For example, a link to a PDF document might look like this `<a href="annual_report.pdf">`, while a link to a zip archive could look like this: `<a href="tutorials.zip">`. In each case, the specific file type is identified by an extension at the end of the URL—.pdf or .zip.

CSS provides an attribute selector that lets you find attributes that end with specific information. So to create a style for links to PDF files, use this selector:

```
a[href$='.pdf']
```

`$=` means "ends in," so this selector means "select all links whose `href` attribute ends in .pdf." You can create similar styles for other types of files as well:

```
a[href$='.zip'] /* zip archive */
a[href$='.doc'] /* Word document */
```

You'll see examples of this technique in the tutorial on page 307.

Tutorial: Styling Links

In this tutorial, you'll style links in a variety of ways, like adding rollovers and background graphics.

To get started, download the tutorial files from this book's companion website at *https://github.com/sawmac/css_mm_4e*. Click the tutorial link and download the files. All the files are enclosed in a zip archive, so you need to unzip them first. The files for this tutorial are contained inside the *09* folder.

Basic Link Formatting

1. **Launch a web browser and open the file *09→links→links.html*.**

   This page contains a variety of links (circled in Figure 9-9) that point to other pages on the site, links to pages on other websites, and an email address. Start by changing the color of the links on this page.

2. **Open *links.html* in a text editor and place your cursor between the opening and closing `<style>` tags.**

   The page already has an external style sheet attached to it with some basic formatting, plus the `<style>` tags for an internal style sheet.

NOTE For this exercise, you'll put the styles in an internal style sheet for easy coding and previewing. But if you were creating the CSS for an entire site, you'd place the styles into a separate CSS file.

3. **Add a new style to the internal style sheet:**

```
<style>
a {
 color: #207EBF;
}
</style>
```

This style is about as generic as it gets. It will apply to all <a> tags on the page. It's a good place to start, since it sets up the overall look of links for the page. You'll add more styles that will let you pinpoint links in specific areas of the page. Now, time to remove that boring old underline beneath the link.

4. **Add** `text-decoration: none;` **to the style you just created.**

This removes the underline, but also makes the link less visible on the page. Remember, you should always do something to make links stand out and seem clickable to your site's visitors.

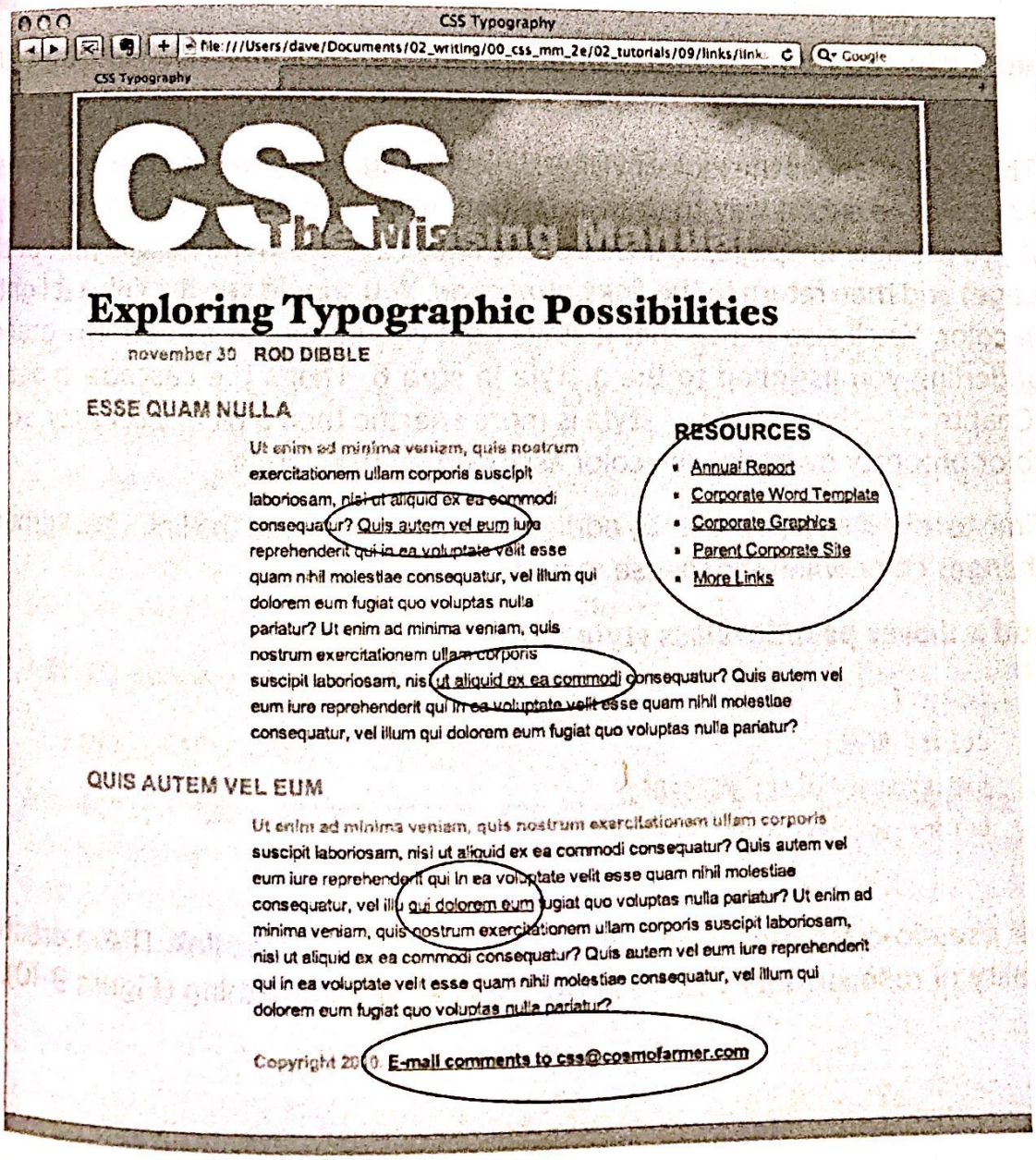

FIGURE 9-9

Here's a basic web page with links in their standard browser configuration—underlined and blue (or purple, if they're links to previously visited pages). In this case, some links point to other pages on the site, some point to other sites, and one is an email address. In this tutorial, you'll style each of these links differently.

5. **Add** `font-weight: bold;` **to the** `a` **style.**

   Now links appear in bold (other text may appear bold, too). Next you'll replace the underline, but you'll do it a bit more creatively, using a border instead of the `text-decoration` property.

6. **Add a** `border` **declaration to the style, so it looks like this:**

```
a {
  color: #207EBF;
  text-decoration: none;
  font-weight: bold;
  border-bottom: 2px solid #F60;
}
```

   The links really stand out, and using a border instead of the normal underline applied to links lets you change the line's color, size, and style (Figure 9-10, left). Now you'll change the look of visited links.

7. **Add a** `:visited` **pseudo-class style for visited links:**

```
a:visited {
 color: #6E97BF;
}
```

   This style changes the look of visited links to a lighter, grayer shade of the main link color—a subtle way to draw attention away from an already visited page. If you preview the page, click one of the links (try one in the middle part of the page) and then return to the *links.html* page. You should see the link get lighter in color. You'll also notice that it stays bold and continues to have the orange underline you assigned to the a style in step 6. That's the cascade in action (Chapter 5)—the `a:visited` style is more specific than a plain a selector, so its color property overrides the color assigned by the a style.

   Time to take it a step further by adding a rollover effect, so the link's background changes color when the mouse moves over it.

8. **Add a** `:hover` **pseudo-class style to the style sheet:**

```
a:hover {
  color: #FFF;
  background-color: #6E97BF;
  border-bottom-color: #6E97BF;
}
```

   This pseudo-class applies only when the mouse is over the link. The interactive quality of rollovers lets visitors know the link does something (Figure 9-10).

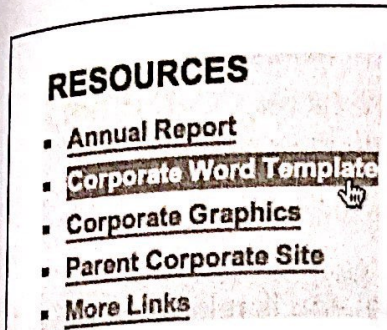

FIGURE 9-10
With a couple of styles, you can change the look of any link. With the `:hover` *pseudo-class, you can even switch to a different style when the mouse moves over the link.*

Adding a Background Image to a Link

The email link at the bottom of the page looks no different than the other links on the page (Figure 9-11, top). You have other plans for that `mailto:` link, however. Since it points to an email address, clicking it doesn't take a visitor to another page, but instead launches an email program. To provide a visual cue emphasizing this point, you'll add a cute little email icon.

1. **Add a descendant selector to the internal style sheet of the *links.html* file:**

```
a[href^="mailto:"] {
  color: #666666;
  border: none;
  background: url(images/email.gif) no-repeat left center;
}
```

This is an advanced attribute selector, which selects any links that begin with `mailto:` (in other words, it selects email links). The `border: none` setting removes the underline defined by the a style you created in step 6—you're going for a subtle look here. The `background` property adds an image on the left edge of the link. Finally, the `no-repeat` value forces the graphic to appear just a single time. Trouble is, the graphic lies directly underneath the link, so it's hard to read the text (circled in the middle image in Figure 9-11).

2. **Add 20 pixels of left padding to the `attribute` style you just created:**

```
padding-left: 20px;
```

Remember that padding adjusts the space between content and its border. So adding some left padding moves the text over 20 pixels but leaves the background in place. One last touch: Move the entire link a little away from the copyright notice.

3. **Add 10 pixels of left margin to the style, so it finally ends up like this:**

```
a[href^="mailto:"] {
  color: #666666;
  border: none;
  background: url(images/email.gif) no-repeat left center;
  padding-left: 20px;
  margin-left: 10px;
}
```

This small visual adjustment makes it clear that the icon is related to the link and not part of the copyright notice (Figure 9-11, bottom).

Highlighting Different Links

At times you may want to indicate that a link points to another website. In this way, you can give your visitors a visual clue that there's additional information elsewhere on the Internet or warn them that they'll exit your site if they click the link. Also, you may want to identify links that point to downloadable files or other non-web-page documents.

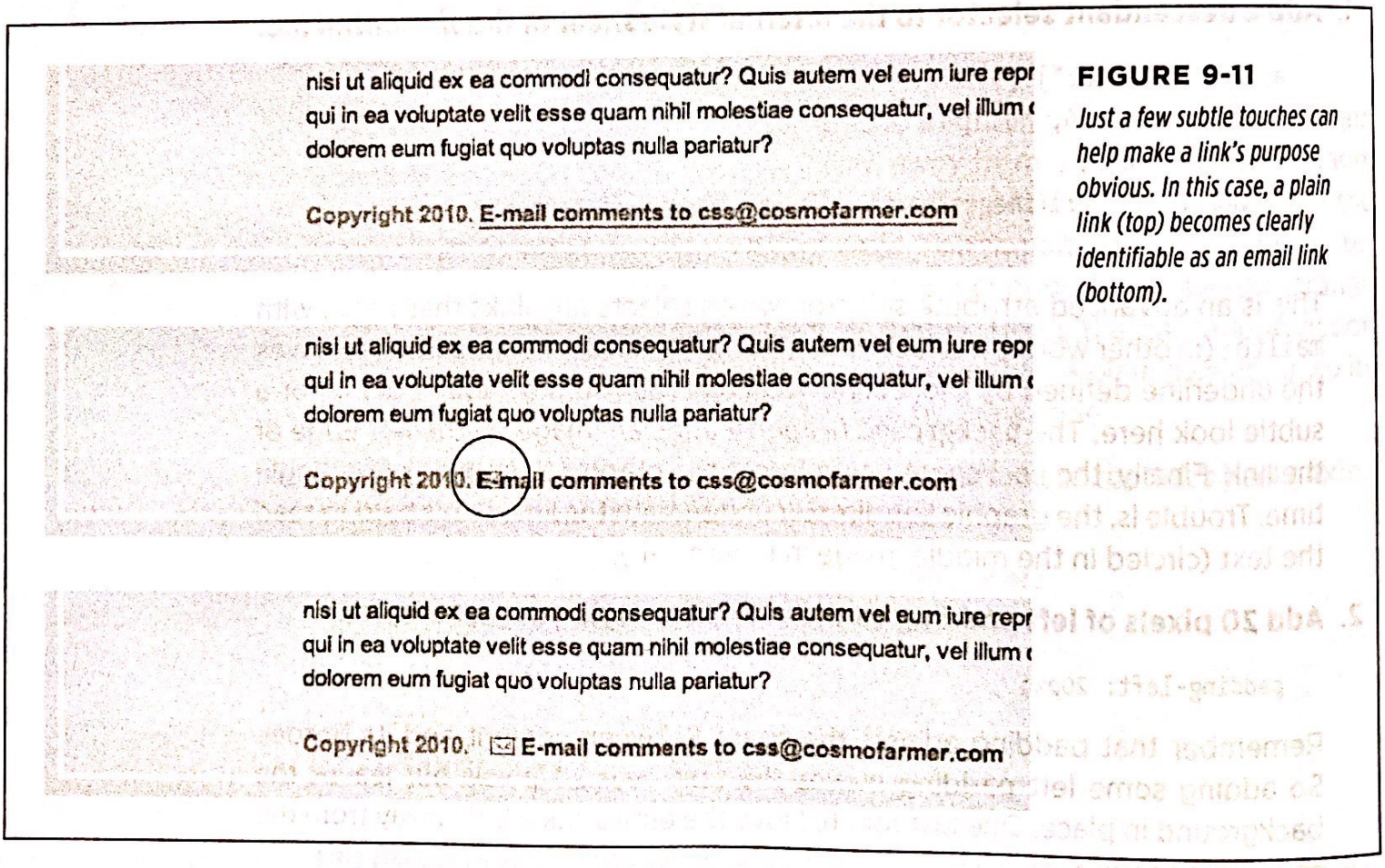

FIGURE 9-11
Just a few subtle touches can help make a link's purpose obvious. In this case, a plain link (top) becomes clearly identifiable as an email link (bottom).

On the web page you're working on, the right-hand "Resources" sidebar contains different types of links that you'll highlight with icons—a different icon for each type of link. First, you'll set up a basic style that applies to all of those links.

1. **Add this style to the *links.html* internal style sheet:**

```
.resources a {
  border-bottom: none;
}
```

Since all of the links you want to format are inside a div with the class resources, the descendant selector `.resources a` targets just those links. This style gets rid of the underline that the generic link style added.

Next, you'll add an icon next to external links.

2. **Add another style at the end of the *links.html* internal style sheet:**

```
.resources a[href*='://'] {
  background: url(images/globe.png) no-repeat right top;
}
```

This is a descendant selector style that uses the advanced attribute selector discussed on page 59. Basically, it targets any link that contains *://* (but only those that are also inside the element with the class `resources`). As with the email link style you created earlier, this style adds a background image. It places the image at the right side of the link.

However, this style has a similar problem as the email link style—the image sits underneath the link's text. Fortunately, the solution is the same—just add some padding to move the image out of the way of the text. In this case, though, instead of adding left padding, you'll add right padding (since the icon appears on the right side of the link). In addition, since every link in the resources box will have a similarly sized icon, you can save some code by adding the padding to the `.resources a` style you created in Step 1.

3. **Edit the `.resources a` style so that it looks like this:**

```
.resources a {
  border-bottom: none;
  padding-right: 22px;
}
```

If you save the page and preview it in a web browser, you'll see small globe icons to the right of the bottom two links in the sidebar. Time to format the other links.

4. **Add three more styles to the internal style sheet:**

```
.resources a[href$='.pdf'] {
  background: url(images/acrobat.png) no-repeat right top;
}
.resources a[href$='.zip'] {
  background: url(images/zip.png) no-repeat right top;
}
.resources a[href$='.doc'] {
  background: url(images/word.png) no-repeat right top;
}
```

These three styles look at how the href attribute ends; identify links to either Adobe Reader files (.pdf), zip archives (.zip), or Word documents (.doc); and assign a different icon in each case.

5. **Finally, add a hover state for the resources links:**

```
.resources a:hover {
  color: #000;
  background-color: rgba(255,255,255,.8);
}
```

This style both changes the color of the text and adds a background color (see Figure 9-12).

You can find a finished version of this tutorial in the *09_finished/links/links.html* file.

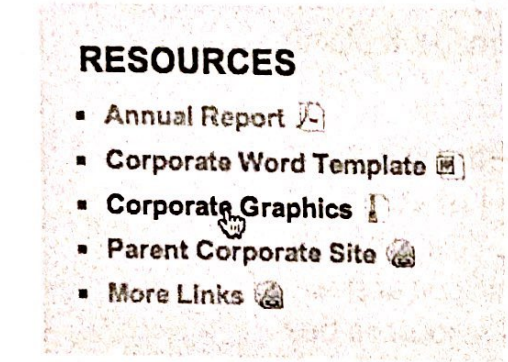

FIGURE 9-12
Using advanced attribute selectors, you can easily identify and style different types of links—external links and links to PDF files, Word docs, and zip files.

Tutorial: Creating a Navigation Bar

In this exercise, you'll turn a plain old list of links into a spectacular navigation bar, complete with rollover effects and a "You are here" button effect.

1. **In a text editor, open *09→nav_bar→nav_bar.html*.**

   As you can see, there's not much to this file yet. There's an internal style sheet with the basic reset styles discussed on page 109, and one rule setting up some basic properties for the <body> tag. The HTML consists of an unordered list with six links. It looks like example #1 in Figure 9-13. Your first step is to add some HTML so you can target your CSS to format the links in this list.

2. **Locate the opening <ul> tag and add class="mainNav" to it, so it looks like this:**

```
<ul class="mainNav">
```

   The class attribute identifies this list as the main navigation area. Use this class to build descendant selectors to format only these links—and not just any old link on the page.

3. **Below the body style in the internal style sheet, add a new style:**

```
.mainNav {
  margin: 0;
  padding: 0;
  list-style: none;
}
```

This style applies only to a tag with a class of `mainNav`—in this case, the `<ul>` tag. It removes the indent and bullets that browsers apply to unordered lists, as shown in #2 in Figure 9-13. Next, you'll start formatting the links.

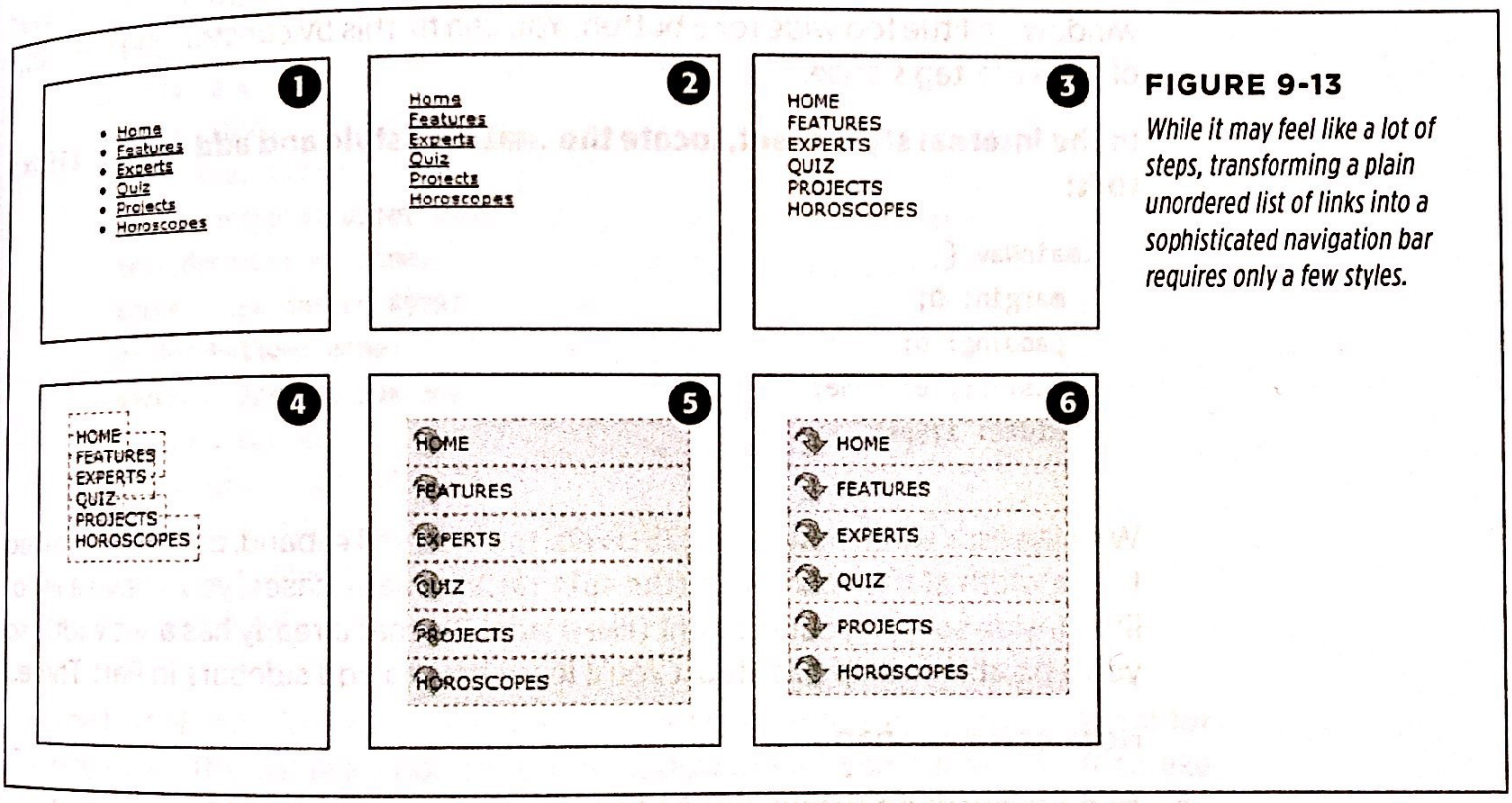

FIGURE 9-13
While it may feel like a lot of steps, transforming a plain unordered list of links into a sophisticated navigation bar requires only a few styles.

4. **Add a descendant selector to format the links in the list:**

```
.mainNav a {
  color: #000;
  font-size: 11px;
  text-transform: uppercase;
  text-decoration: none;
}
```

This style defines the basic text formatting for the links. It sets the color and font size, makes all letters uppercase, and removes the line usually found underneath links (#3 in Figure 9-13). Now start making the links look like buttons.

5. **To the** `.mainNav a` **style, add the following** `border` **and** `padding` **properties:**

```
border: 1px dashed #999;
padding: 7px 5px;
```

If you preview the file now, you'll see a few problems (#4 in Figure 9-13): The borders overlap and the boxes aren't the same width. That's because the <a> tag is an inline element, so the width of the box is just as wide as the text in the link. In addition, top and bottom padding don't add any height to inline boxes, so the borders overlap. (See page 192 for a discussion of inline boxes.) You can fix these problems by changing how a browser displays these links.

6. **Add** `display: block;` **to the** `.mainNav a` **style.**

   You've changed the basic display of the <a> tag so it acts like a paragraph or other block-level element, with the links neatly stacked one on top of the other. The only problem now is that they also extend the full length of the browser window—a little too wide for a button. You can fix this by constraining the width of the <ul> tag's style.

7. **In the internal style sheet, locate the** `.mainNav` **style and add** `width: 175px;` **to it:**

```
.mainNav {
  margin: 0;
  padding: 0;
  list-style: none;
  width: 175px;
}
```

   With the list's width now set to 175 pixels, the links still expand, but they're limited to the width of their container (the <ul> tag). In many cases, you'll have a list of links inside some layout element (like a sidebar) that already has a set width, so you'll be able to skip this step. (You'll learn how to add sidebars in Part Three.)

   Now for the fun part.

8. **Add background properties to the** `.mainNav a` **style, like so:**

```
.mainNav a {
  color: #000;
  font-size: 11px;
  text-transform: uppercase;
  text-decoration: none;
  border: 1px dashed #999;
  padding: 7px 5px;
  display: block;
  background-color: #E7E7E7;
  background-image: url(images/nav.png);
  background-repeat: no-repeat;
  background-position: 0 2px;
}
```

These lines add a gray background color to the links and a non-repeating image at the left edge of each button (#5 in Figure 9-13). You still have a couple of things to fix: The link text overlaps the icon, and the border between each button is 2 pixels thick. (Technically, the borders are still just 1 pixel thick, but the bottom and top borders of adjoining links are creating a 2-pixel line.)

TIP Using the background shorthand property, you can write the code in step 8 like this: `background: #E7E7E7 url(images/nav.png) no-repeat 0 2px;`.

9. **Remove the bottom border and adjust the padding for the `.mainNav a` style, so it looks like this:**

```
.mainNav a {
  color: #000;
  font-size: 11px;
  text-transform: uppercase;
  text-decoration: none;
  border: 1px dashed #999;
  border-bottom: none;
  padding: 7px 5px 7px 30px;
  display: block;
  background-color: #E7E7E7;
  background-image: url(images/nav.png);
  background-repeat: no-repeat;
  background-position: 0 2px;
}
```

The text of each link sits clear of the icon and the borders look great...except for one thing. The last link's bottom border is now gone. (Sometimes CSS feels like two steps forward, one step back!) But you have a few ways to fix this snafu. One way is to apply a bottom border to the `<ul>` tag containing the list of links. (Since there's no padding on that tag, there's no space separating the top of the `<ul>` from the top of that first link.) But another way is to use the `:last-of-type` pseudo-class (page 65). You just need to select the link that's inside the last list item in the navigation bar, and give it a bottom border.

10. **Add the following style between `.mainNav` and the `.mainNav a` styles:**

```
.mainNav li:last-of-type a {
  border-bottom: 1px dashed #999;
}
```

This descendant selector styles the link (a) that's inside the last list item (`li:last-of-type`) or the navigation list (`.mainNav`).

There you have it: a basic navigation bar using borders, padding, background color, and images (#6 in Figure 9-13).

Adding Rollovers and Creating "You Are Here" Links

Now it's time to add some interactive and advanced features to this nav bar. First, you'll add a rollover effect to the buttons in your main navigation bar. That way, the buttons change to show your visitor which button she's about to click.

It's also considerate to let your visitor know which page of your site she's on. Using the same HTML nav bar you already have, you can make this bit of interactivity happen automatically. You simply make the button's format change to match the page's section. Sounds simple, but it does require a little planning and setup, as you'll see in the following steps.

The rollover effect is easy, so get that out of the way first:

1. **In the *nav_bar.html* file, add the following style to the end of the style sheet:**

```
.mainNav a:hover {
  font-weight: bold;
  background-color: #B2F511;
  background-position: 3px 50%;
}
```

This style sets the button's hover state. It makes the text inside the button bold, and changes the background color to a vibrant green. In addition, it uses the CSS sprites technique discussed on page 298. The same image is used as in step 8 on page 310—however, that image actually holds three different icons (see Figure 9-14). In this case, the image is centered within the button, displaying the middle icon in the file.

Now, moving the mouse over any of the buttons instantly changes its look. (Open the page in your web browser and try it yourself.)

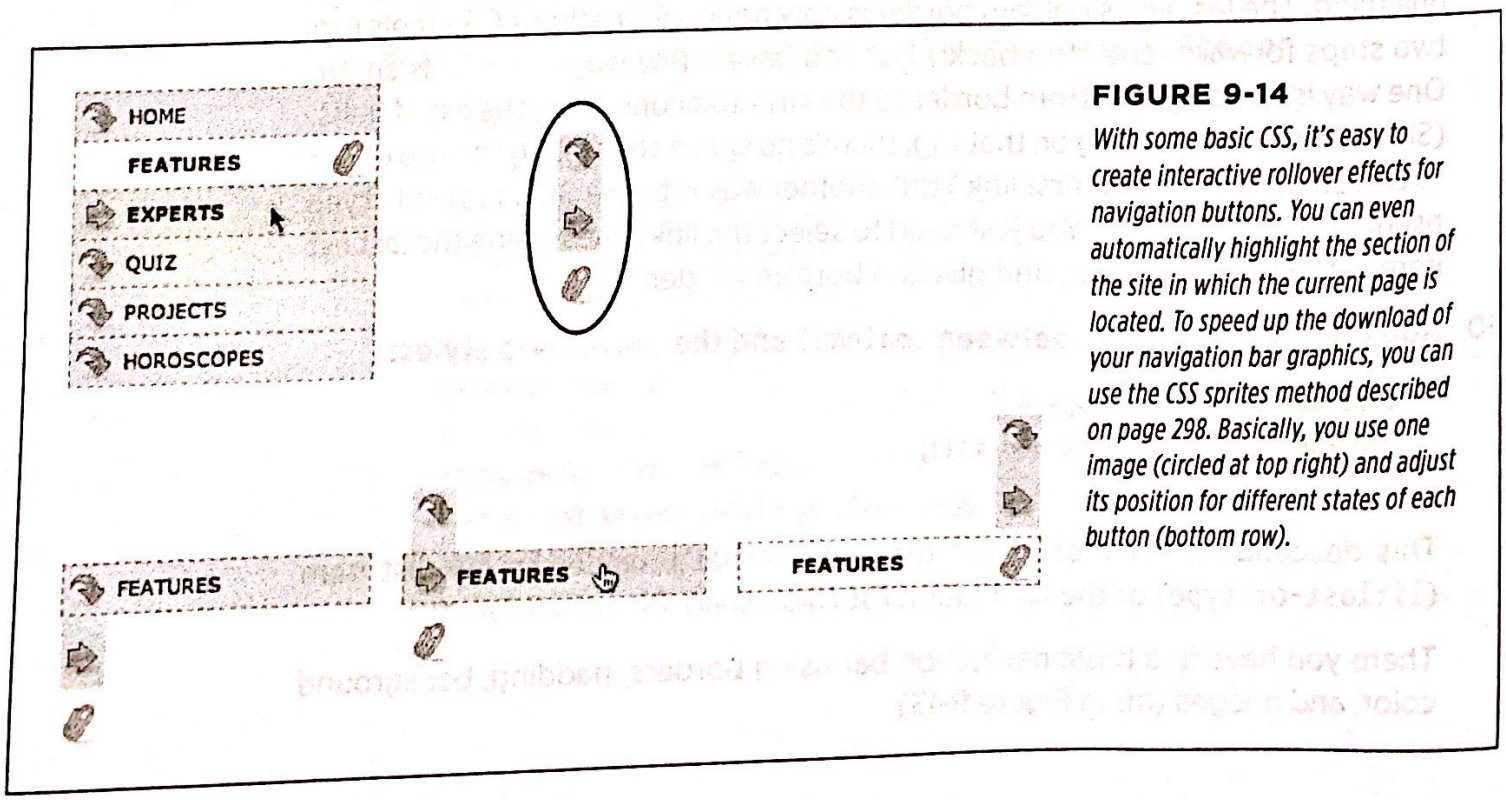

FIGURE 9-14

With some basic CSS, it's easy to create interactive rollover effects for navigation buttons. You can even automatically highlight the section of the site in which the current page is located. To speed up the download of your navigation bar graphics, you can use the CSS sprites method described on page 298. Basically, you use one image (circled at top right) and adjust its position for different states of each button (bottom row).

Next, make your navigation bar more informative by highlighting the button that matches the section in which the page is located. To do so, you need to identify two things in the nav bar's HTML: the section a page belongs to and the section each link points to. For this example, assume that the page you're working on is the home page.

NOTE Alternatively, you can create a class style that changes the appearance of a link and apply it to the link representing the page's section. For a horoscope page, you'd apply the class to the Horoscope link in the nav bar: `<a href="/horoscopes/" class="current">Horoscopes</a>`.

2. **Locate the `<body>` tag, and then add `class="home"`, like so:**

```
<body class="home">
```

Now that you know what section this page belongs to, you can use a descendant selector to create special CSS rules that apply only to tags on pages within the Features section. Next, you need to identify the section each link applies to, which you accomplish by adding some classes to those links.

3. **In the nav bar's HTML code, locate the Home link, and then add `class="homeLink"` so the tag looks like this:**

```
<a href="/index.html" class="homeLink">Home</a>
```

This class identifies this link, providing the information you need to create a style that applies only to that link.

You need to add a class to the other links in the navigation bar as well.

4. **Repeat step 3 for each of the other links using the following classes:** `featureLink`, `expertLink`, `quizLink`, `projectLink`, and `horoscopeLink`.

You're done with the HTML part of this exercise. Now it's time to create some CSS. Because you've provided classes to identify the different links, it's easy to create a descendant selector to highlight the Home link.

5. **Add another style to the page's style sheet:**

```
.home .homeLink {
  background-color: #FFFFFF;
  background-position: 97% 100%;
  padding-right: 15px;
  padding-left: 30px;
  font-weight: bold;
}
```

You've seen all these properties before. Again, you're using the CSS sprites method to adjust the position of the background image. This time, the image is moved over to the right 97 percent (that is, the point 97 percent across the image is matched up with the point 97 percent across the button), and the bottom of the image is placed at the bottom of the button. In other words, it displays the

icon at the bottom of the image (see Figure 9-14). See page 240 for a discussion of how percentage values work with background-images.

The most interesting part is the selector—`.home .homeLink`. It's a very specific selector that applies only to a link with a class of `homeLink` that's *also* inside a `<body>` tag with a class of `home`. If you change the class of the page to `quiz`, for example, the link to the Home page is no longer highlighted.

Preview the page in a browser to see the effect: The Home link now has a white background and a paper clip icon. To make this work for the other links, you need to expand this selector a little...OK, make that a *lot*.

6. **Edit the selector for the style you just added, like so:**

```
.home .homeLink,
.feature .featureLink,
.expert .expertLink,
.quiz .quizLink,
.project .projectLink,
.horoscope .horoscopeLink{
  background-color: #FFFFFF;
  background-position: 97% 100%;
  padding-right: 15px;
  padding-left: 30px;
  font-weight: bold;
}
```

Yes, that's a lot of CSS. But your setup work here has a big payoff. This style now applies to every link in the nav bar, but only under certain conditions, which is exactly how you want it to behave. When you change the `class` attribute of the `<body>` tag to `quiz`, the link to the Quiz gets highlighted instead of the link to the Features section. Time to take your work for a test drive.

NOTE This long-winded selector is an example of the group selector discussed on page 49.

7. **Change the `class` attribute of the `<body>` tag to `feature` like this:**

```
<body class="feature">
```

Preview the page, and wham! The Feature link is now highlighted with a white background and a paper clip icon (Figure 9-14). The secret at this point is to just change the class in the `<body>` tag to indicate which section of the site a page belongs to. For a horoscope page, change the class to `class="horoscope"` in the `<body>` tag.

NOTE Ready to take this design further? Try adding a rollover effect to complement the style you created in step 6. (Hint: Use the `:hover` pseudo-class as part of the selector like this: `.quiz .quizLink:hover`.) Also try adding a different graphic for the Home link. (You have a *home.png* file in the images folder to use.)

To see the completed version of this navigation bar, see the file *09_finished→nav_bar→nav_bar_vertical.html*.

From Vertical to Horizontal

Suppose you want a horizontal navigation bar that sits at the top of the page. No problem—you did most of the hard work in the last part of this tutorial. Just modify that page a little to spread the buttons along a single line. (You'll use the *nav_bar.html* file you just completed, so if you want to keep the vertical nav bar, then save a copy of the file before proceeding.)

1. **Make sure you've completed all the steps above to create the vertical navigation bar, and have the file *nav_bar.html* open in your text editor.**

   Now you'll see how easy it is to change the orientation of a navigation bar. Start by cleaning up some of the work you already did. You need to remove the width you set for the `<ul>` tag in step 7 on page 310. That width prevented the nav buttons from spanning the entire length of the page. But since the `<ul>` needs to spread out much wider to contain the side-by-side buttons, this width has to go.

2. **Find the `.mainNav` style, and then remove the `width: 175px;` declaration.**

   And now it's time for the big secret of horizontal nav bars—placing the buttons side by side.

3. **Add a new style to your style sheet (directly below the `.mainNav` style is a good spot):**

```
.mainNav li {
  float: left;
  width: 12em;
}
```

   This style applies to the `<li>` tag (the list items that hold each link). The first declaration floats the tag to the left. In this way, each `<li>` tag attempts to wrap around to the right side of the previous `<li>` tag. Also, setting the width of the `<li>` tag defines the width of each button. Here, a value of 12 ems provides enough space to contain the longest link name—Horoscopes. When you're working with longer links, you need to increase this value.

   If you preview the page now, you'll see the basics are complete. All that's left are some cosmetic enhancements (see the circled areas of #1 in Figure 9-15). Currently, there's no bottom border below the buttons, and the border where buttons touch doubles up (because the right border of one button combines with the left border of the link next to it), so you'll fix those issues next.

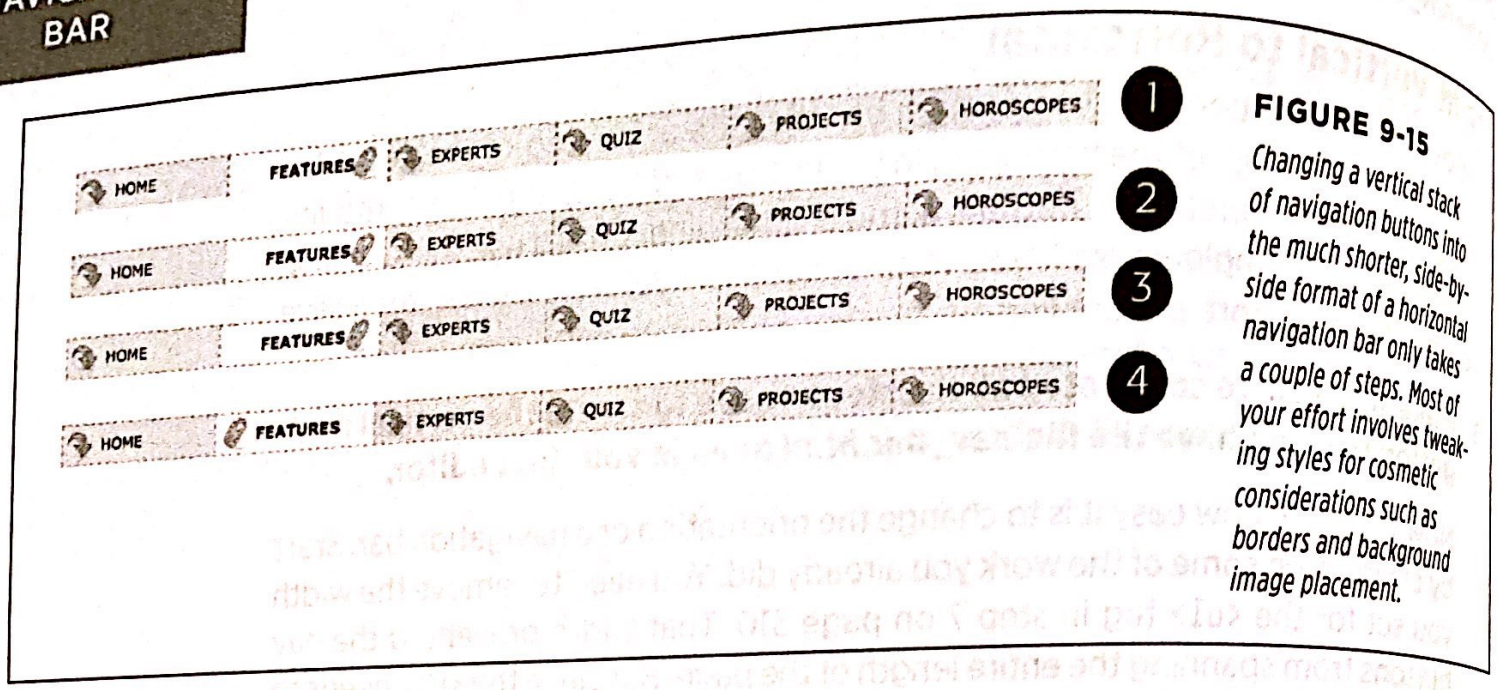

FIGURE 9-15
Changing a vertical stack of navigation buttons into the much shorter, side-by-side format of a horizontal navigation bar only takes a couple of steps. Most of your effort involves tweaking styles for cosmetic considerations such as borders and background image placement.

4. **In the** `.mainNav a` **style, change** `border-bottom: none;` **to** `border-right: none;`.

   This change removes the right border so the borders don't double up between buttons, and at the same time restores the border to the bottom of each button. But now the border on the right side of the last navigation button is missing (#2 in Figure 9-15). You can use the `:last-of-type` selector to fix that—you already have that style in place from the previous part of this tutorial.

5. **Change the** `border-bottom` **property of the** `.mainNav li:last-of-type a` **style to** `border-right` **like this:**

```
.mainNav li:last-of-type a {
  border-right: 1px dashed #999;
}
```

   This change adds a right border, but only to the last link in the navigation bar (#3 in Figure 9-15). Finally, that paper clip aligned to the right edge of the "You are here" button looks odd (#3 in Figure 9-15). You'll switch its position to the left edge of the button.

6. **Locate the "You are here" style you created in step 6 on page 314. (It's the one with the crazy, long-winded selector.) Change its background position from** `97% 100%` **to** `3px 100%`**. The style should now look like this:**

```
.home .homeLink,
.feature .featureLink,
.expert .expertLink,
.quiz .quizLink,
.project .projectLink,
.horoscope .horoscopeLink
{
```

```
    background-color: #FFFFFF;
    background-position: 3px 100%;
    padding-right: 15px;
    padding-left: 30px;
    font-weight: bold;
}
```

Preview the page, and you'll find a fully functional horizontal navigation bar (#4 in Figure 9-15).

To see the finished version, open the file *09_finished→nav_bar→nav_bar_horizontal.html*.

NOTE You may want to center the text inside each button. If so, you need to do two things: Add `text-align: center;` to the `.mainNav a` style, and adjust that style's `left-padding` until the text looks absolutely centered.
