The Essential Guide to Dreamweaver CS4 with CSS, Ajax, and PHP

David Powers

The Essential Guide to Dreamweaver CS4 with CSS, Ajax, and PHP

Copyright © 2009 by David Powers

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

ISBN-13 (pbk): 978-1-4302-1610-0

ISBN-13 (electronic): 978-1-4302-1611-7

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Trademarked names may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, we use the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com.

For information on translations, please contact Apress directly at 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705. Phone 510-549-5930, fax 510-549-5939, e-mail info@apress.com, or visit www.apress.com.

Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Special Bulk Sales–eBook Licensing web page at http://www.apress.com/info/bulksales.

The information in this book is distributed on an "as is" basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

The source code for this book is freely available to readers at www.friendsofed.com in the Downloads section.

Credits

Lead Editor
Ben Renow-Clarke

Technical Reviewer
Tom Muck

Editorial Board
Clay Andres, Steve Anglin, Mark Beckner, Ewan Buckingham, Tony Campbell, Gary Cornell, Jonathan Gennick, Michelle Lowman, Matthew Moodie, Jeffrey Pepper, Frank Pohlmann, Ben Renow-Clarke, Dominic Shakeshaft, Matt Wade, Tom Welsh

Project Manager
Beth Christmas

Copy Editors
Kim Wimpsett, Marilyn Smith

Associate Production Director
Kari Brooks-Copony

Production Editor
Kelly Winquist

Compositor
Molly Sharp

Proofreader
Patrick Vincent

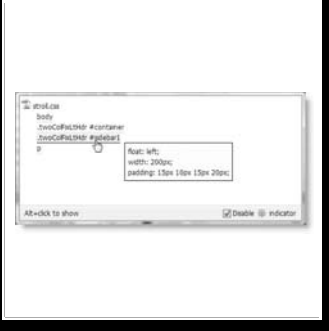
Indexer
Julie Grady

Artist
April Milne

Interior and Cover Designer
Kurt Krames

Manufacturing Director
Tom Debolski

6 BUILDING SITE NAVIGATION WITH THE SPRY MENU BAR



Efficient and attractive navigation is an important element in every website. The Spry menu bar combines CSS and JavaScript (using Spry, Adobe's implementation of Ajax) to create a flexible menu with flyout submenus that remains accessible even if JavaScript is turned off. In essence, it's an unordered list with optional nested lists for submenus. It comes in two versions: horizontal and vertical. Figure 6-1 shows what the horizontal version of the Spry menu bar looks like when integrated into the page built in the previous chapter.

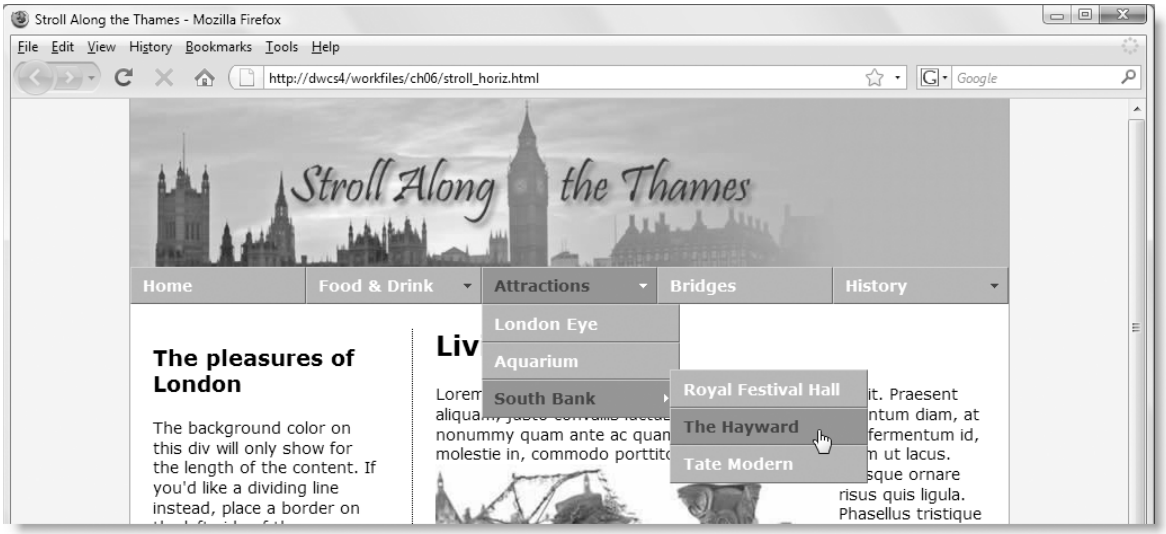


Figure 6-1. You can easily integrate the Spry menu bar into a page by making a few adjustments to the CSS.

Although you can insert a Spry menu bar in seconds, the downside is that styling it requires a good understanding of CSS. Knowing which style rules to change—and which to leave alone—presents more of a challenge. This process has been made considerably easier in Dreamweaver CS4 by the introduction of Live view and Code Navigator.

In this chapter, you'll learn about the following:

- The structure of the Spry menu bar
- How to insert and remove a Spry menu bar
- The style rules that control a Spry menu bar
- How to customize a Spry menu bar using Live view and Code Navigator

By the end of the chapter, you'll be able to transform the rather bland default design of a menu bar into something much more elegant like the menu in Figure 6-1. Because the Spry menu bar is styled with CSS, this chapter assumes you're familiar with the CSS Styles panel, which was described in detail in Chapter 4.

Examining the structure of a Spry menu bar

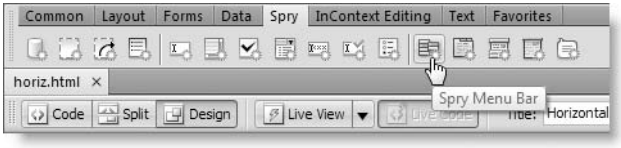
The Spry menu bar relies on external files to control the way it looks and works, so you must always save your page in a Dreamweaver site (see Chapter 2 for how to define a site) before attempting to insert a menu bar. If you forget, Dreamweaver tells you to save your page and opens the Save As dialog box.

The best way to understand how a Spry menu bar works is to launch Dreamweaver and start experimenting.

Inserting a horizontal menu bar

This brief exercise takes you through the steps of inserting a horizontal Spry menu bar in a new page.

1. Create a blank HTML page in Dreamweaver by selecting File ➤ New. In the New Document dialog box, select Blank Page, HTML for Page Type, and <none> for Layout. Make sure that no style sheets are listed under Attach CSS file before clicking Create. Alternatively, just select New ➤ HTML from the welcome screen. Save the file as `horiz.html` in `workfiles/ch06`.
2. Select the Spry tab on the Insert bar, and click the Spry Menu Bar button (it's the fifth from the right), as shown in the following screenshot:



3. This opens the Spry Menu Bar dialog box. There are just two options: Horizontal and Vertical. Select Horizontal, and click OK.
4. Dreamweaver inserts a horizontal Spry menu bar at the top of the page, as shown in Figure 6-2. Like all Spry widgets, the menu bar is surrounded in Design view by a turquoise border and a tab at the top-left corner. The tab tells you what type of widget it is, followed by the widget's `id` attribute. Dreamweaver calls the first menu bar on a page `MenuBar1`. The next one is `MenuBar2`, and so on. This means you can have as many menu bars on a page as you want (don't go mad—think of usability).

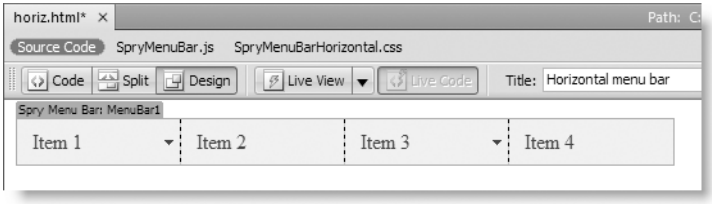


Figure 6-2. The Spry menu bar is given basic styling ready for you to customize.

5. Notice that the Related Files toolbar lists two files: `SpryMenuBar.js` and `SpryMenuBarHorizontal.css`. Until you save the page, these are temporary files. You can verify this by switching to Code view and inspecting the code in the <head> of the page, as in Figure 6-3.

```
5  <title>Untitled Document</title>
6  <script src="file:///C|/Users/Work/AppData/Roaming/Adobe/Dreamweaver
   CS4/en_US/Configuration/Temp/Assets/eam8AB9.tmp/SpryMenuBar.js" type="text/javascript"></script>
7  <link href="file:///C|/Users/Work/AppData/Roaming/Adobe/Dreamweaver
   CS4/en_US/Configuration/Temp/Assets/eam8AB9.tmp/SpryMenuBarHorizontal.css" rel="stylesheet" type=
   "text/css" />
8  </head>
```

Figure 6-3. Dreamweaver uses temporary files for the style sheet and JavaScript until you save the page.

As you can see on lines 6 and 7 in Figure 6-3, the links to the external JavaScript file and style sheet point to a temporary folder on my local hard disk. Therefore, it's essential to save the file before doing anything else. Otherwise, any changes you make to the style sheet are likely to be lost. Moreover, the menu won't work unless the files are saved to your Dreamweaver site.

6. Save `horiz.html`. If this is the first time you have inserted a Spry menu bar in the current site, you are prompted to save the dependent files (see Figure 6-4).

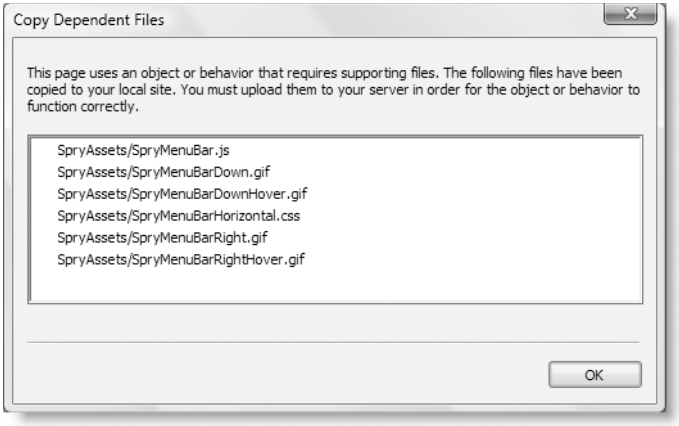


Figure 6-4. The Spry files need to be copied to your site the first time you insert a menu bar.

As you can see in Figure 6-4, four images are also copied to your site. These are the navigation arrows that appear on submenus. When you click OK, Dreamweaver locates the files in the Spry assets folder. By default, this is called `SpryAssets`, but you can specify a different location in your site definition (see "Setting other site options" in Chapter 2). Once the files have been copied to the Spry assets folder, they are shared with further instances of the menu bar in the same site.

7. Click the Live View button in the Document toolbar, and run your mouse pointer over the menu bar. As you can see in Figure 6-5, you already have a menu bar ready to customize.

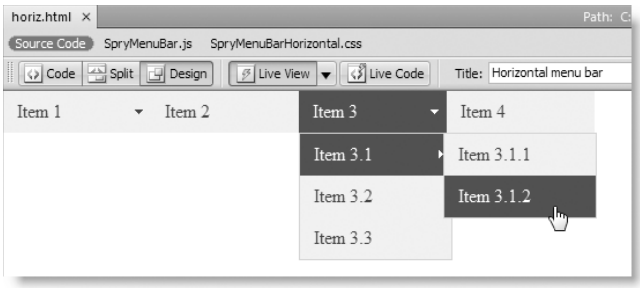
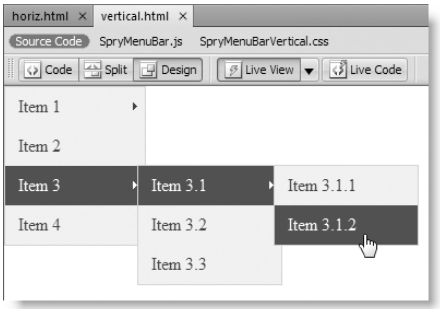


Figure 6-5.
The structure and styling of the default menu bar are fully customizable.

Inserting a vertical menu bar is the same. The only differences are that you select the Vertical radio button in step 3 and Dreamweaver inserts `SpryMenuBarVertical.css` instead of the style sheet for the horizontal menu bar. The menu items in the vertical menu bar are stacked vertically, and the first-level submenus fly out to the right rather than beneath the main menu, as shown in the screenshot alongside.



Looking at the menu bar's structure

The Spry menu bar is a series of nested unordered lists (`<ul>`) styled with CSS to look like a series of buttons. The submenu flyouts are controlled by JavaScript. You can see the underlying structure of the menu either by switching to Code view or by toggling the Turn Styles Off/On button in the Property inspector. (If you can't see the button, click the Spry Menu Bar tab at the top left of the menu bar.) Figure 6-6 shows the horizontal menu bar in `horiz.html`, but the structure is identical in a `vertical` menu. The different look and functionality are controlled entirely by JavaScript and CSS.

Click this tab to display the menu bar details in the Property inspector

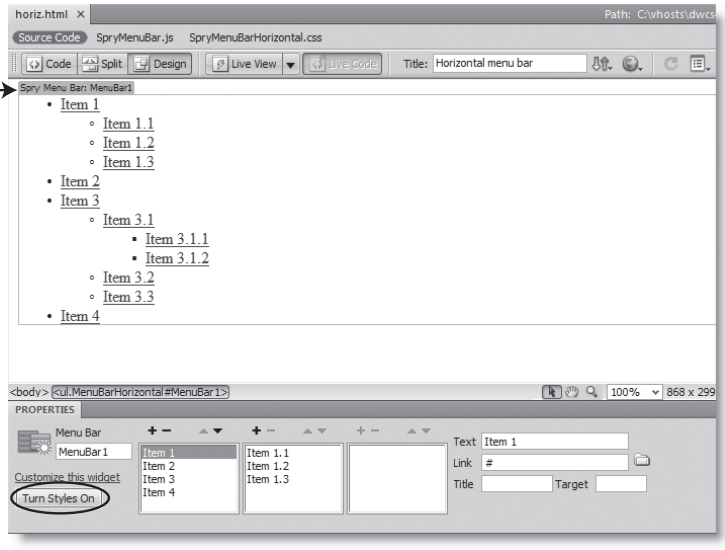


Figure 6-6. When styles are turned off, you can see the underlying list structure of the menu bar.

Figure 6-7 shows the same menu in Code view. The code on lines 6 and 7 link the external JavaScript file and style sheet to the page. The unordered list that contains the menu bar is on lines 11–33. The block of JavaScript at the foot of the page on lines 34–38 initializes the JavaScript object that controls the menu.

```
5  <title>Horizontal menu bar</title>
6  <script src="../../SpryAssets/SpryMenuBar.js" type="text/javascript"></script>
7  <link href="../../SpryAssets/SpryMenuBarHorizontal_stroll.css" rel="stylesheet" type="text/css" />
8  </head>
9
10 <body>
11 <ul id="MenuBar1" class="MenuBarHorizontal">
12   <li><a class="MenuBarItemSubmenu" href="#">Item 1</a>
13       <ul>
14         <li><a href="#">Item 1.1</a></li>
15         <li><a href="#">Item 1.2</a></li>
16         <li><a href="#">Item 1.3</a></li>
17       </ul>
18   </li>
19   <li><a href="#">Item 2</a></li>
20   <li><a class="MenuBarItemSubmenu" href="#">Item 3</a>
21       <ul>
22         <li><a class="MenuBarItemSubmenu" href="#">Item 3.1</a>
23             <ul>
24               <li><a href="#">Item 3.1.1</a></li>
25               <li><a href="#">Item 3.1.2</a></li>
26             </ul>
27         </li>
28         <li><a href="#">Item 3.2</a></li>
29         <li><a href="#">Item 3.3</a></li>
30       </ul>
31   </li>
32   <li><a href="#">Item 4</a></li>
33 </ul>
34 <script type="text/javascript">
35 <!--
36 var MenuBar1 = new Spry.Widget.MenuBar("MenuBar1", {imgDown:"../../SpryAssets/SpryMenuBarDownHover.gif",
   imgRight:"../../SpryAssets/SpryMenuBarRightHover.gif"});
37 //-->
38   </script>
39 </body>
40 </html>
```

Figure 6-7. The scripts at the top and bottom of the page control the menu's look and action.

When you add further content to the page, this initialization script remains just before the closing </body> tag. If a menu stops working, you should always check that you haven't deleted the initialization script by mistake. If you have, you need to go back and reinsert the menu from scratch.

Editing a menu bar

Since the menu bar is just a series of nested unordered lists, you can turn off the styles, as shown in Figure 6-6, and edit the menu directly in Design view. However, it's much more convenient to do it in the Property inspector. Place your cursor anywhere inside the menu bar, and click the Spry Menu Bar tab at the top left to display the menu bar details in the Property inspector.

The three columns in the center of the Property inspector show the menu hierarchy, with the top level on the left. When you select an item in this column, the middle one displays the contents of the related submenu. The right column displays the next level down from whatever is selected in the middle one.

To edit a menu item, highlight it, and fill in the fields on the right of the Property inspector as follows:

- Text: This is the label you want to appear on the menu button.
- Link: This is the page to which you want to link. Either type the file name directly into the field or click the folder icon to the right of the field to browse to the target page.
- Title: This adds a `title` attribute to the link. Most browsers display this as a tooltip. It can also improve accessibility for visually impaired people using a screen reader by describing the link's destination more fully.
- Target: This adds a `target` attribute to the link. This was originally designed for use with frames. A value of `_blank` opens the linked page in a new browser window. Although there are sometimes legitimate reasons for opening a new window, it's rarely justified to do so from a site's navigation menu. The practice of using `target="_blank"` provokes a lot of heated debate, so use with care.

To add an item, click the plus (+) button at the top of the relevant column. To delete an item, select it and click the minus (–) button. You can also change the order of items by highlighting them and using the up and down arrows at the top of each column.

As Figure 6-6 shows, the Property inspector lets you work on two levels of submenus. To create a submenu at a deeper level, insert another nested list either by turning off styles as shown in Figure 6-6 or editing directly in Code view. Two levels of submenus should be sufficient for most purposes. If your menus require more levels, it's probably time to rethink the structure of your site.

After editing a menu bar, select one of the items in the left column before moving to another part of the page. If you forget to do this, the submenus remain exposed in Design view, preventing you from working on the underlying part of the page.

If this happens, position your cursor inside any part of the menu bar, and select the Spry Menu Bar tab at the top left. This populates the Property inspector with the menu bar details again. You can then select an item in the left column to hide the submenus.

Maintaining accessibility with the Spry menu bar

The Spry menu bar is much more accessible than the JavaScript pop-up menus in old versions of Dreamweaver, because the underlying structure and links are written in HTML, rather than being obscured in JavaScript that search engines can't follow. However, it's important to realize that JavaScript still controls the submenu flyouts. If someone visits your site with JavaScript disabled or an ancient browser that can't understand the Spry code, the only parts of the menu that remain accessible are the top-level items.

This means you should always link the top-level items to a real page and not just use dummy links to act as triggers for the submenus. So, for instance, if anyone clicks Attractions in the menu shown in Figure 6-1, it should link to an introductory page leading to that section. Unless you do so, some visitors may never be able to get to the pages about London Eye and so on.

Customizing the styles

Although the color scheme of the default style sheets isn't exactly inspiring, the structural layout has been carefully thought out, so you don't need to change many properties to achieve a rapid transformation of the menu bar. Select `SpryMenuBarHorizontal.css` in the Related Files toolbar, and take a look at how the style rules are divided into the following sections:

- **Layout Information**: This controls the structure, such as font size and menu widths.
- **Design Information**: This styles the color scheme and borders.
- **Submenu Indication**: The rules in this section control the display of the arrows that indicate the existence of a submenu. Change these only if you need to adjust the submenu arrows.
- **Browser Hacks**: These rules deal with bugs in Internet Explorer. You should leave them alone.

The style sheet for a vertical menu (`SpryMenuBarVertical.css`) is laid out in the same way. In fact, both style sheets contain almost identical rules, although the names of the CSS selectors reflect the orientation of the menu. The horizontal bar uses the class `MenuBarHorizontal`, and the vertical one uses `MenuBarVertical`.

Customizing the CSS rules requires a good understanding of the hierarchy within the menu bar's nested lists. The entire menu is contained in an unordered list, so all selectors begin with either `ul.MenuBarHorizontal` or `ul.MenuBarVertical`. Submenus are also unordered lists nested within the main one, so rules that apply to submenus all use `ul.MenuBarHorizontal ul` or `ul.MenuBarVertical ul`. However, the same rules apply to links in both the main menu and the submenu, so they use `ul.MenuBarHorizontal a` or `ul.MenuBarVertical a`.

There are a few other things to note:

- All the measurements use relative units (ems and percentages).
- The width of the horizontal menu is set to `auto`, but the vertical menu has a fixed width of `8em`.
- The width of the menu items in both versions is fixed at `8em`; submenus are `8.2em`.

Changing the menu width

The use of ems for the width of the menu and submenu items makes the menu bar very fluid. As explained in Chapter 4, an em is a typographical term that has been borrowed by CSS to mean the height of the specified font. So, the width expands and contracts depending on the size chosen for the font. For a fixed layout, such as that used in `stroll.html` in the previous chapter, you need to change all instances of `8em` and `8.2em` in the Layout Information section to a fixed width in pixels.

Changing colors

All colors are defined in the Design Information section of the style sheet. Changing them is simply a matter of substituting the existing hexadecimal numbers for `background-color` and `color` in the relevant style rules. The default colors are light gray (`#EEE`) for the background and dark gray (`#333`) for the text of menu items in their normal state, and navy blue (`#33C`) for the background and white (`#FFF`) for the text of items in a rollover state.

The menu bar uses JavaScript to assign a class dynamically to the links when the mouse pointer moves over them. For some reason, Adobe has put the selectors for this dynamic class in a separate style rule, which duplicates the `a:hover` and `a:focus` rules like this:

```
ul.MenuBarHorizontal a:hover, ul.MenuBarHorizontal a:focus
{
  background-color: #33C;
  color: #FFF;
}
ul.MenuBarHorizontal a.MenuBarItemHover, ul.MenuBarHorizontal
a.MenuBarItemSubmenuHover, ul.MenuBarHorizontal a.MenuBarSubmenuVisible
{
  background-color: #33C;
  color: #FFF;
}
```

Since both rules contain the same properties and values, it's simpler to combine the selectors like this:

```
ul.MenuBarHorizontal a:hover, ul.MenuBarHorizontal a:focus,
ul.MenuBarHorizontal a.MenuBarItemHover, ul.MenuBarHorizontal
a.MenuBarItemSubmenuHover, ul.MenuBarHorizontal a.MenuBarSubmenuVisible
{
  background-color: #33C;
  color: #FFF;
}
```

Don't forget to add a comma after `a:focus` in the first line of the selector. Otherwise, it won't work. The equivalent rules for the vertical menu bar are identical, except for the class name `MenuBarVertical`.

Adding borders

By default, a light gray border is added to the outer edge of the submenu containers in both the horizontal and vertical menu bars. In addition, the vertical menu bar has the same border around the entire menu. Change the following rules to alter the menu and submenu borders:

```
ul.MenuBarHorizontal ul
ul.MenuBarVertical
ul.MenuBarVertical ul
```

Individual menu items don't have any borders, so the menu looks seamless. If you want to give your menu a more button-like feel, apply a border to the following rules:

```
ul.MenuBarHorizontal a
ul.MenuBarVertical a
```

The links in the menu bar are styled to display as a block and have no fixed width. Consequently, applying a border to the link style has the advantage of surrounding the individual menu items without affecting either height or width. You'll see how this is done when inserting a menu bar into `stroll.html`.

Changing the font

The `font-size` property is set to `100%` in two separate rules: `ul.MenuBarHorizontal` and `ul.MenuBarHorizontal li` (`ul.MenuBarVertical` and `ul.MenuBarVertical li`). Change the wrong one and you get the mysterious shrinking text shown in Figure 6-8.

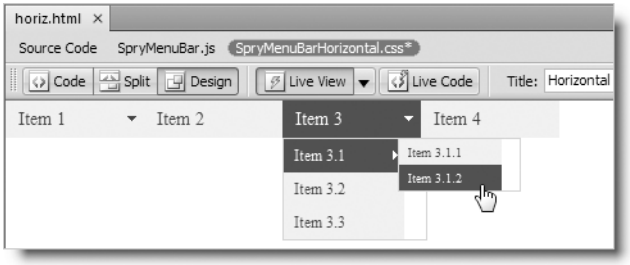


Figure 6-8. The text gets progressively smaller if you change font-size in the li selector.

The style rules that affect the size of the text in the horizontal menu bar are `ul.MenuBarHorizontal` and `ul.MenuBarHorizontal li`. Both of them set `font-size` to `100%`. The shrinking text in Figure 6-8 was caused by changing `font-size` in `ul.MenuBarHorizontal li` to `85%`.

Although this reduces the text in the main menu items to 85 percent of its original size, the nesting of the submenus results in the first-level submenu being displayed at 85 percent × 85 percent—in other words, 72.25 percent. The second-level submenu is further reduced by another 85 percent—resulting in 61.4 percent.

To prevent this happening, leave the `ul.MenuBar Horizontal li` selector at `100%`, and change only the first one. The following rules produce a consistent text size:

```
ul.MenuBarHorizontal
{
  font-size: 85%;
}
ul.MenuBarHorizontal li
{
  font-size: 100%;
}
```

The rules for the vertical menu bar are identical, except for the class name `MenuBarVertical`.

If you decide to use pixels instead of percentages, it doesn't matter which rule you change. You should be aware, however, that using pixels for fonts can cause accessibility problems for people with poor eyesight. Many designers mistakenly believe that using pixels for font sizes "locks" their design. It doesn't, because all browsers—apart from Internet Explorer for Windows—permit users to adjust font sizes by default, and Internet Explorer's accessibility features have an option to ignore font sizes. If a change in font size causes your page to fall apart, you need to rethink your design criteria—fast.

Styling a Spry menu bar

If you're completely at home editing style sheets in Code view, the preceding sections tell you all you need to know about customizing the CSS for a Spry menu bar. With the Related Files feature enabled, you can edit the style rules in the Code view section of Split view and monitor the changes by refreshing the Design view section of the Document window. I'm going to devote the rest of the chapter to showing you how to add a horizontal menu bar to `stroll.html`, the CSS layout that you styled in the previous chapter. You can see the finished menu in Figure 6-1 at the beginning of this chapter.

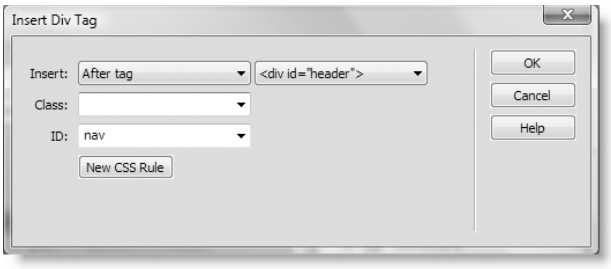
To wrap or not to wrap, that is the question . . .

When I started working with Spry, my first instinct was to use the horizontal Spry menu bar without a `<div>`. After all, it's an unordered list, which is a block element, and it has its own ID, so it should be possible to drop one into a page without the need for a wrapper. After much experimentation, though, I discovered that a horizontal menu bar in a fixed-width design like `stroll.html` behaves unpredictably in some older browsers unless you wrap it in a `<div>` with both a specified width and height. The height is needed because all the menu items are floated.

Inserting the horizontal menu bar

Continue working with your files from the previous chapter. Alternatively, copy `stroll_horiz_start.html` and `stroll_horiz_start.css` from `examples/ch06` to `workfiles/ch06`. Rename the files `stroll_horiz.html` and `stroll_horiz.css`, and update any links when prompted.

1. With `stroll_horiz.html` open in the Document window, select the Common or Layout tab of the Insert bar, and click the Insert Div Tag button.
2. You're going to insert the `<div>` to accommodate the Spry menu beneath the header `<div>`, so use the following settings in the Insert Div Tag dialog box (refer to Chapter 3 if you need to refresh your memory about inserting a `<div>`):



3. Click the New CSS Rule button at the bottom of the Insert Div Tag dialog box.
4. The New CSS Rule dialog box should automatically be populated with the correct values for Selector Type and Selector Name, but you should make sure that Rule Definition is set to the existing style sheet (`stroll.css`). Check that your values are the same as in Figure 6-9, and then click OK to accept.

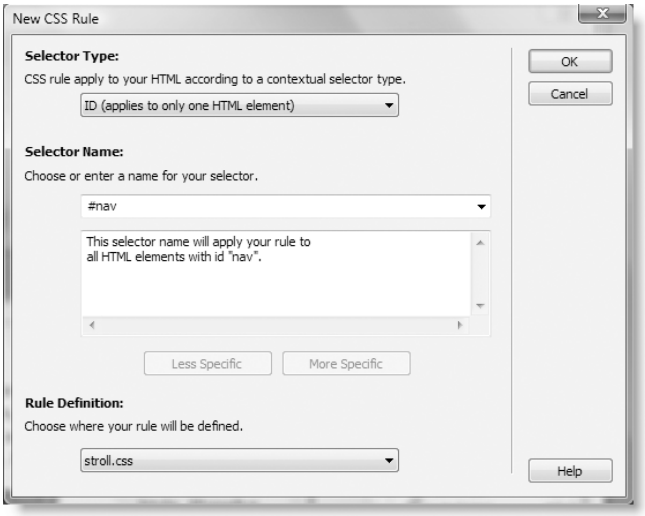


Figure 6-9. The settings for creating the style rule for the nav

5. In the CSS Rule Definition dialog box, select the Box category, where you need to set the width and height for the nav `<div>`. The width is easy; it needs to be the same as the `container` `<div>` that wraps the page content: `780px`. This ensures the menu bar will remain snugly in the `<div>`, even if the user increases the font size. I calculated the height by adding together the top and bottom padding (`0.5em` each) for the links in the menu bar. The `font-size` property is set to `100%`, which is the same as `1em`. That makes `2em`. After testing, I decided to add an extra `.2em` to make sure everything fits. Using relative units for the height ensures that the `<div>` expands vertically to accommodate enlarged text.

   Set Width to 780px and Height to 2.2em. Click OK to save the rule. This returns you to the Insert Div Tag dialog box. Click OK again to close it. You should now have a `<div>` with some placeholder text in it just beneath the header, as shown here:

When using a vertical menu bar, you can simply drop it into a sidebar, which provides the necessary wrapper. Unless the sidebar is particularly wide, there is no need for a separate `<div>` *for the menu itself.*

6. You need to get rid of the placeholder text for the nav `<div>`. Normally, pressing Delete when the text is highlighted is sufficient. However, it's a good idea to open Split view to make sure that it's only the text between the `<div>` tags that is selected.

   If necessary, go into Code view to adjust the selection, and press Delete. Make sure your cursor is between the empty `<div>` tags.

7. Click the Spry Menu Bar button on the Spry tab of the Insert bar (it's also on the Layout tab), and insert a horizontal menu bar.

8. Save `stroll_horiz.html`. If you did the other exercises earlier in this chapter, Dreamweaver won't prompt you to save dependent files this time, because they have already been copied to the Spry assets folder. The top of your page should now look like Figure 6-10.

Figure 6-10. The Spry menu bar needs to be restyled to fit into the rest of the page.

9. Select the Spry Menu Bar tab, and edit the menu items as described in "Editing a menu bar" earlier in the chapter. If you want to follow my structure, here it is:

```
Home
Food & Drink
    Restaurants
    Bars
Attractions
    London Eye
    Aquarium
    South Bank
        Royal Festival Hall
        Hayward Gallery
        Tate Modern
Bridges
History
    St Paul's Cathedral
    Tower of London
    Houses of Parliament
```

In a live website, you need to create links to real pages, but for the purposes of the example page, I have left the value of each link as # so the menu bar displays correctly, even though it doesn't link to other pages.

10. If you have used the same menu structure as me, you'll see that a long item, such as Food & Drink, wraps onto a second line. This pushes the sidebar across to the right, as shown in Figure 6-11.

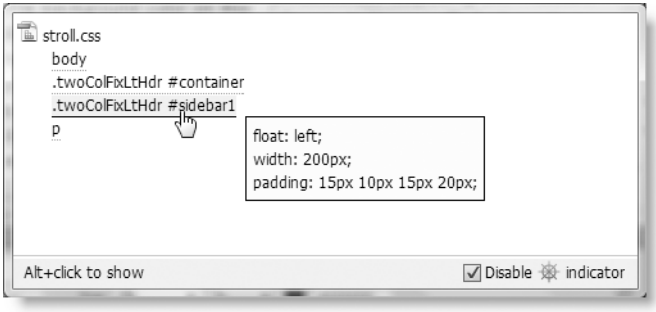
Figure 6-11. Long menu items prevent subsequent floated elements from moving to the left of the viewport.

To rectify this, you need to add `clear: left` to the sidebar's style block. This is necessary because both the menu buttons and the sidebar are floated, so the sidebar tries to move into the nearest available space. By adding `clear: left`, the

sidebar is instructed to move below any previously floated elements and go to the left side of its parent, the `container <div>`.

Hold down the Alt key on Windows or Opt+Cmd on a Mac, and click anywhere inside the sidebar to launch the Code Navigator.

The Code Navigator displays the names of all CSS selectors that apply to the section of the page that you clicked. Each selector displayed in the Code Navigator is a link that shows the existing properties and values. Click the link for the `.twoColFixLtHdr #sidebar1` selector, as shown here:



11. This opens the style sheet in Split view, with your cursor in the selected rule ready to edit it. Add clear: left; to the `.twoColFixLtHdr #sidebar1` rule, and press F5 to refresh Design view. You should see the sidebar move to its correct position, as shown in Figure 6-12.

Figure 6-12. Pressing F5 after editing a style rule in Split view lets you see the effect instantly.

12. Select File ➤ Save All to save the changes to both `stroll_horiz.html` and `stroll.css`.

Customizing the design

If you test the page in Live view or a browser, you have a navigation bar, but it looks pretty ugly. The challenge is to customize the CSS to fit the rest of the page. This involves two basic stages, namely:

- Adjusting the width of the menu items so that the navigation bar stretches the full width of the page. The submenus have a separate width so that needs to be adjusted too.
- Changing the colors so they blend in harmoniously with the rest of the page. At the same time, you can add borders to the items to make them look more like buttons.

First, though, you need to do a little housekeeping with the menu's style sheet.

Editing the default selectors

All style rules exclusive to the menu bar are in `SpryMenuBarHorizontal.css` in the Spry assets folder. Since this is common to all horizontal menu bars, it's a good idea to give it a different name. Also, as I mentioned earlier, the rollover colors for the submenus are declared in a separate style rule. Unless you want them to be different from the main menu items, it makes life a little easier to combine them into a single rule.

1. Select `SpryMenuBarHorizontal.css` in the `SpryAssets` folder in the Files panel, and gently click the file name once to open its name for editing (alternatively, press F2, or right-click and select Edit ➤ Rename from the context menu). Change the style sheet's name to `SpryMenuBarHorizontal_stroll.css`, and press Enter/Return.

   Accept the option to update links when prompted. This updates the link to the external style sheet in both `horiz.html` and `stroll_horiz.html`. Since `horiz.html` was only a test page, it doesn't matter on this occasion, but in a working project, you need to check which links are being updated.

2. Open `stroll_horiz.html` in Code view. As explained in the previous chapter, Dreamweaver adds new style sheets immediately before the closing `</head>` tag. This puts the styles in `SpryMenuBarHorizontal_stroll.css` lower in the cascade than the style rules in the conditional comments. Although nothing is likely to clash, it's good practice to cut and paste the link above the conditional comments. Place it immediately after the link to `stroll_horiz.css`.

3. Select `SpryMenuBarHorizontal_stroll.css` in the Related Files toolbar, and locate the following section:

```
 98  /* Menu items that have mouse over or focus have a blue background and white text */
 99  ul.MenuBarHorizontal a:hover, ul.MenuBarHorizontal a:focus
100  {
101      background-color: #33C;
102      color: #FFF;
103  }
104  /* Menu items that are open with submenus are set to MenuBarItemHover with a blue background and
     white text */
105  ul.MenuBarHorizontal a.MenuBarItemHover, ul.MenuBarHorizontal a.MenuBarItemSubmenuHover,
     ul.MenuBarHorizontal a.MenuBarSubmenuVisible
106  {
107      background-color: #33C;
108      color: #FFF;
109  }
```

4. Insert a comma after `a:focus` at the end of line 99 in the preceding screenshot, and delete lines 100–104 (use the line numbers in the screenshot only as a guide; it's the code that matters). You should end up with this:

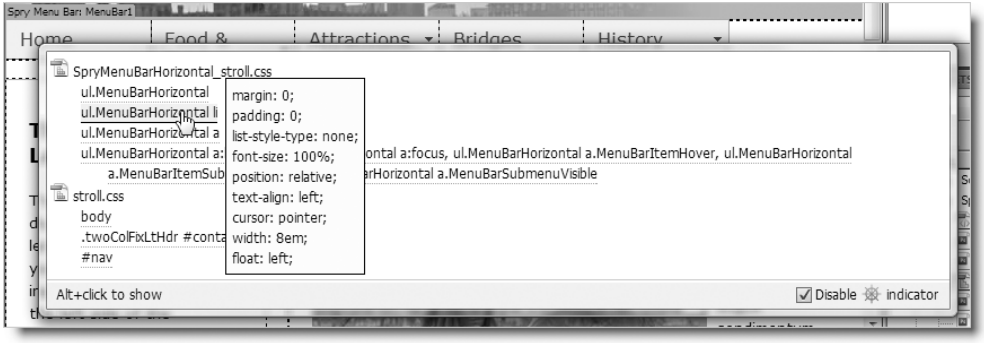
```
98  /* Menu items that have mouse over or focus have a blue background and white text */
99  ul.MenuBarHorizontal a:hover, ul.MenuBarHorizontal a:focus,
100 ul.MenuBarHorizontal a.MenuBarItemHover, ul.MenuBarHorizontal a.MenuBarItemSubmenuHover,
    ul.MenuBarHorizontal a.MenuBarSubmenuVisible
101 {
102     background-color: #33C;
103     color: #FFF;
104 }
```

Save `SpryMenuBarHorizontal_stroll.css`, and switch back to Design view.

Customizing the menu bar: setting widths

The default width of the menu items is `8em`, but this is a fixed width design, so you need to adjust the menu bar to fit. There are five top-level items, and the width of the `container` `<div>` is 780 pixels. A quick calculation reveals that dividing 780 by 5 equals 156. So that's the width each item needs to be.

1. The menu bar is a styled unordered list, so the width of each item is controlled by the `<li>` element. Hold down the Alt key (Opt+Cmd on a Mac), and click anywhere in the menu bar to open the Code Navigator. Move the mouse pointer over the link for the `ul.MenuBarHorizontal li` rule, as shown in the following screenshot:

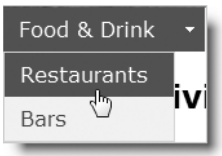


As you can see, the `width` property is set to `8em`. Click the link for the `ul.MenuBarHorizontal li` rule to open the style sheet in Split view.

2. Dreamweaver should position your cursor at the beginning of the `ul.MenuBarHorizontal li` rule. Change the value of the `width` property to `156px`, and press F5 to refresh Design view. You should now see the menu fits neatly across the page, as shown in Figure 6-13 (you might need to switch back to Design view if your monitor isn't wide enough to see the effect in Split view).

Figure 6-13. Giving the <li> elements a fixed pixel width matches the width of the container .

3. Click the Live View button, and move your mouse pointer over the menu bar until you trigger one of the submenus. As you can see from the screenshot alongside, the submenus are now narrower than the main menu items.



The width of the submenus is controlled independently. Some of my submenu items are long, so let's make the submenus 20px wider than the main items, in other words, 176px. With your mouse still over one of the submenu items, hold down the Alt key (or Opt+Cmd), and click to activate the Code Navigator. This is where Live view and the Code Navigator really shine. This time, the Code Navigator also detects the style rules that affect the submenus in their hover state. As you mouse over each selector in the Code Navigator, the properties and values of each style rule are displayed as a tooltip.

Go down each one in turn until you find the rule that sets the width for the submenus. It's `ul.MenuBarHorizontal ul`, as shown in Figure 6-14. Click it to edit the rule in Split view.

Figure 6-14. In Live view, the Code Navigator detects style rules that affect dynamically generated elements.

4. Change the `width` property of `ul.MenuBarHorizontal ul` to `176px`.

5. Press F5 to refresh Design view, and move the mouse pointer over the menu to trigger one of the submenus again. Contrary to what you might expect, the submenu items are still too narrow. If you look closely, you'll see that there's a thin gray border surrounding the whole submenu (I have deliberately exaggerated the border in the screenshot alongside to make it stand out more on the printed page). It's the correct width, but the individual submenu items are still their original width.

6. Open the Code Navigator again, and inspect the style rules until you find one that defines the `width` property as `8.2em` (it's `ul.MenuBarHorizontal ul li`). Click the link to edit the style rule, and change the value of `width` to `176px`.

7. Press F5 to refresh Design view, and test the submenu again. The individual items should now be the correct width.

8. I'm going to add a border to each menu item, so let's get rid of the default border around the submenus. Trigger one of the submenus in Design view, and open the Code Navigator again. The border is defined in the `ul.MenuBarHorizontal ul` selector. As explained earlier in this chapter, the rules in the style sheet are divided into sections covering colors, layout, and so on. Consequently, there are two rules for `ul.MenuBarHorizontal ul`. Click the second one, which contains the `border` property, and change the value in the style sheet to `border: none;`.

   Keep `stroll_horiz.html` open, because I'll show you how to adjust the colors next.

Customizing the menu bar: changing colors and fonts

The main colors of the Spry menu bar are controlled in style rules applied to the links. These instructions assume you have edited the menu bar style sheet as described in "Editing the default selectors."

1. Make sure Live view is still active, hold down the Alt key (or Opt+Cmd), and click any menu item that *doesn't* lead to a submenu. In `stroll_horiz.html`, this means Home or Bridges. The colors of the menu items are defined in the `ul.MenuBarHorizontal a` rule. This is a descendant selector that applies to all links in the menu bar. Click the selector in the Code Navigator so you can edit the rule's properties in Split view.

2. Change `background-color` from `#EEE` to `#A3AAC6` (mauve) and `color` from `#333` to `#FFF` (white). Press F5 to see the colors updated in Design view.

3. Things are beginning to look better, but let's add a border around the links to make them look like buttons. Add the following properties and values to the `ul.MenuBarHorizontal a` rule (I'll explain how I arrived at these values later in the chapter):

```
border-left: #C4C9DB 1px solid;
border-top: #C4C9DB 1px solid;
border-right: #565968 1px solid;
border-bottom: #565968 1px solid;
```

4. Press F5 to refresh Design view. The menu links should now look more button-like, but when you pass your mouse over them, the rollover colors need fixing.

   Although you could use the Code Navigator to find the rollover selector, it's a lot quicker to just scroll down in the style rules in Split view, because it's the next rule down (it begins with `ul.MenuBarHorizontal a:hover`).

   Change `background-color` from `#33C` to `#7A85AD` (dark mauve) and `color` from `#FFF` to `#333` (very dark gray). Press F5, and mouse over the menu to see the changes in Design view.

5. There's just one final change: the font would look better if it were bold and slightly smaller. As I explained in "Customizing the styles" earlier in the chapter, the place to change font properties is in the `ul.MenuBarHorizontal` rule. The quickest way to find it in the style sheet is with the Code Navigator, so hold down Alt/Opt+Cmd, and click the menu in Design view. Then click the `ul.MenuBarHorizontal` selector in the Code Navigator.

6. Change the value of `font-size` to `90%`, and add `font-weight: bold;` to the rule.

7. Select File ➤ Save All to save the page and style sheet. Test the page in a browser. You should now have an attractive menu bar as shown in Figure 6-1 at the beginning of this chapter.

   You can check your files against `stroll_horiz.html` in `examples/ch06` and `SpryMenuBarHorizontal_stroll.css` in the `SpryAssets` folder.

Even if the text size is enlarged, the page structure is preserved, and the dark gray rollover text ensures that spillover text remains reasonably legible. Enlarging the text does disrupt the original design of the page, but certain trade-offs are inevitable in web design. The purpose here has been to show you how to customize a Spry menu bar, rather than seek a definitive answer to accessibility issues.

These instructions have concentrated on customizing a horizontal menu bar, but the process is the same for a vertical one. The main difference is that you don't need to wrap a vertical menu bar in a `<div>` of its own. However, if you do decide to use a separate `<div>`, it shouldn't have a fixed height. Otherwise, you may run into display problems if the user enlarges the text in the browser.

Choosing border colors

In the past, it was common to use images to create menu buttons, but that's no longer necessary with CSS. Styling links to display as a block makes the background color fill the full width and height of each link. To give the link a raised effect like a button, all you need to do is put a border around them, using a lighter color for the top and left borders and a darker one for the right and bottom borders.

A neat way of finding the right colors is to create a rectangle in a graphics program like Fireworks, give the rectangle the same color as your buttons, and then apply an inner bevel effect. Figure 6-15 shows how it's done in Fireworks CS4.

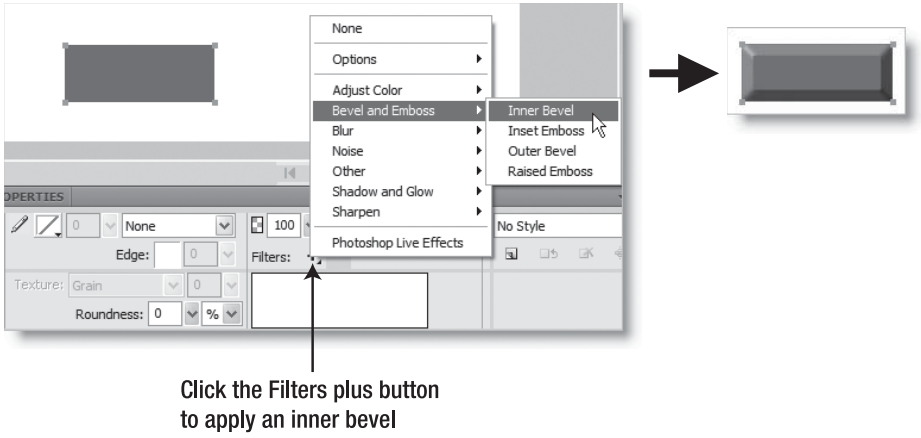


Figure 6-15. Use a graphics program to add a bevel to a block of solid color to find the best colors for CSS borders.

Use an eyedropper tool to find the appropriate colors for the lighter and darker borders, and make a note of the hexadecimal number. In this case, it's probably easier to use the eyedropper tool in your graphics program, but there's a useful trick if you want to copy the color of an object outside Dreamweaver. Adjust the size of the Dreamweaver workspace so that you can see the object, click the color picker, and hold down the mouse button. You can then drag the eyedropper outside Dreamweaver. The color picker in Dreamweaver constantly updates to show the color currently being sampled by the eyedropper. Release the mouse button when you find the color you want.

Removing a menu bar

Removing a menu bar is quite simple: click the Spry Menu Bar tab at the top left of the menu (see Figure 6-6), and press Delete. That's it—not only is the HTML code removed but so too are the links to the external JavaScript file and style sheet, as well as the initialization script at the bottom of the page. However, the dependent files in the Spry assets folder are *not* removed. This ensures they remain accessible to other pages that may rely on them.

Moreover, the links to the external JavaScript file and style sheet are not removed if another instance of the same type of menu exists on the page.

> *It's important to remove menu bars cleanly by selecting the* Spry Menu Bar *tab and pressing Delete. Otherwise, the initialization script shown on lines 34–38 of Figure 6-7 remains in the underlying code and might trigger errors when the page is loaded into a browser.*

Chapter review

Because it's built with HTML and CSS, the Spry menu bar is accessible and search engine–friendly. However, I'm sure that many noncoders will find customizing the CSS an uphill struggle. Instead of creating menu buttons in a graphic environment and letting the software take care of the coding, much more is left up to the designer's individual skill. However, Live view and the Code Navigator make the job considerably easier than it was in Dreamweaver CS3.

The CSS skills required to customize a menu bar are essential for building modern standards-compliant sites. In my own experience, CSS is not something you can pick up overnight, but once the various pieces begin to fall together, progress becomes much more rapid. So if you're struggling, keep at it, and it will all come together in the end. In Chapter 12, I'll show you how to adapt the menu bar and move it to an external file that can be included in all pages on a site, greatly reducing the amount of maintenance required.

The menu bar is just one of many Spry widgets and effects in Dreamweaver. In the next chapter, we'll look at Spry effects, tabbed panels, the accordion, and collapsible panels.