5

Part Five: Dreamweaver CS3 Power

CHAPTER
18

Snippets and Libraries

You've finished the design for your company's new Web site. It looks great and your boss is ecstatic. But you've really only just begun. There are hundreds of pages to build before you launch. And once the site's online, you'll need to make endless updates to keep it fresh and inviting.

This is where Dreamweaver's Snippet and Library features come in, streamlining the sometimes tedious work of building and updating Web pages.

As you build more and more Web pages (and more and more Web sites), you may find yourself creating the same Web page elements over and over again. Many pages of a site may share certain common elements that are always the same: a copyright notice, a navigation bar, or a logo, for example. And you may find yourself frequently using more complex items, such as a pull-down menu listing all the countries your company ships products to, or a particular design you use for photos and their captions.

Recreating the same page elements time after time is tiresome and—thanks to Dreamweaver—unnecessary. Dreamweaver provides two subtly different tools for reusing common page elements: *Snippets* and *Library items*.

Snippets Basics

Snippets aren't fancy or complex, but they sure save time. A snippet is simply a chunk of code that you store away and then plunk into other Web pages. It could be HTML, JavaScript, or any of the other programming languages you may encounter. Dreamweaver comes with hundreds of snippets organized into different folders, like Footers (canned footer designs), Form Elements (useful form parts like pull-down menus), and JavaScript code (programming code for interesting effects like adding a random image to a page).

For example, say you always use the same table design to list the specifications for a product in your company's catalog. Each time you want to create a similar table, you could go through all the same steps to build it—or you could turn that table into a snippet and then, with a simple double-click, add it to page after page of your site.

You keep these code chunks in the Snippets tab of the Files panel (see Figure 18-1). You get to them in any of several ways:

- Choose Window → Snippets.
- Windows people can press Shift-F9. (There's no Mac keyboard shortcut for opening the Snippets tab, but you can create your own if you want, as described on page 733).
- Click the Snippets tab on the Files panel.

Above and beyond Dreamweaver's preinstalled snippets, you can quickly build a collection of your own.

Using Snippets

Snippets come in two varieties: those that are a simple block of code and those that wrap around whatever you've currently selected in the document. For example, in the Text folder of the Snippets tab, you'll find a snippet called Service Mark. Adding this snippet to a page instantly inserts the code *<sup>sm</sup>*, creating a superscript service mark (^{SM}) symbol.

But on occasion you may want to wrap code around something you've already typed. You may want to add an HTML comment to your page (a message that won't appear in a Web browser, but that you can use for helpful notes to yourself or other Web designers). The "Comment, multi-line" snippet (in the Comments folder) can help you quickly add such comments. It wraps whatever you've selected with opening (<!--) and closing HTML comments (-->). Adding an HTML comment is as easy as typing the comment, selecting it, and then inserting this snippet. (This may sound a lot like the Apply Comment button in the Coding toolbar described on page 376. The cool thing about this snippet is that it works in Design view, too, not just Code view.)

Note: Unfortunately, unless the snippet's description (visible in the Snippet Panel's *Description* column) specifies that the snippet wraps, there's no way to tell whether a snippet is intended to wrap around a selection. You either have to try the snippet or open the snippet in editing mode (see page 651) to find out. (While you've got the snippet open, you can add a note to its description indicating its ability, or inability, to wrap.)

To add a snippet to a Web page, click in the document where you want the item to go, or select the object you wish to wrap with a snippet. Then do one of the following:

- Double-click the name of the snippet on the Snippets tab of the Files panel.

- Select the snippet on the Snippets tab, and then click the panel's Insert button.
- Drag the snippet from the panel into the document window. (If the snippet is supposed to wrap a selection, drag the snippet *onto* the selected object.)

Snippets can be used in either Design or Code view (see page 370), but some snippets make sense only in Code view. For example, the JavaScript snippets that come with Dreamweaver typically have to be inserted in the <head> of a page, inside <script> tags. To use them, you must switch to Code view, insert the script tags, and then put the snippets inside.

Tip: To quickly insert a snippet you've recently used, select the snippet from the Insert → Recent Snippets menu. Better yet, create a keyboard shortcut for your favorite snippets and insert them with a quick keystroke as described on page 733.

Snippets simply dump their contents into a document; Dreamweaver doesn't step in to make sure that you're adding the code correctly. Unless you're careful—and have some knowledge of HTML—you may end up adding snippets that make your Web page unviewable. (For advice on how to avoid such pitfalls, see the box on page 653.)

Creating Snippets

Dreamweaver comes with a lot of snippets, and you many have no use for many of them. No problem—it's simple to create snippets of your own. Here's how:

1. **Create and select the code you wish to turn into a snippet.**

   You could, for instance, select a table in Design view, or select the opening and closing <table> tags (as well as all the code between them) in Code view. Or, if you wanted to save a pull-down form menu (see page 414) that took you half an hour to build, just click the form menu in Design view.

   If you make a snippet out of code that isn't visible in Design view, such as a JavaScript program or content that appears in the <head> of the page, you need to switch into Code view first and then select the code.

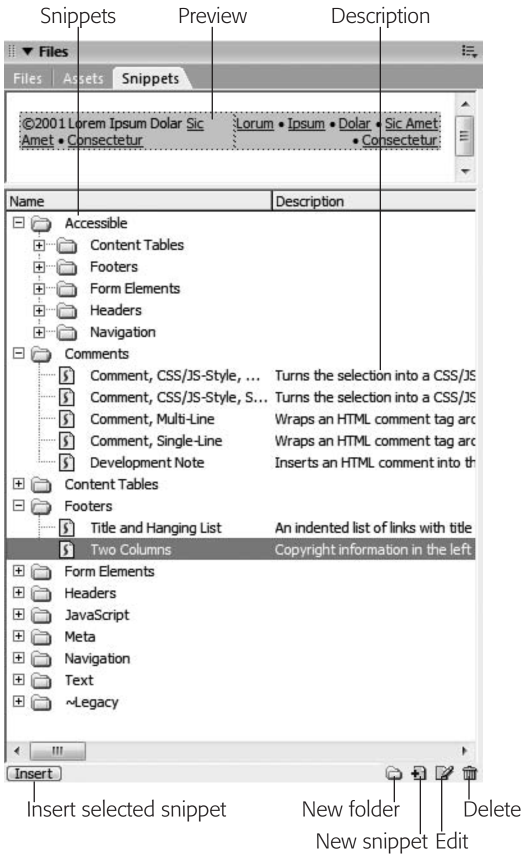
2. **Click the New Snippet button on the Snippets tab (Figure 18-1).**

   The Snippet window appears (Figure 18-2), displaying the code you selected in the Insert field.

Tip: If you skip step 1 and just click the New Snippet button, you can either type the code or paste a previously copied selection into the Insert box (see step 6).

3. **Title the snippet.**

   The name you type in the Name field appears in the Snippets tab. Make sure to give it an easily understood name.



©2001 Lorem Ipsum Dolar Sic Amet • Consectetur Lorum • Ipsum • Dolar • Sic Amet • Consectetur

Name	Description
Accessible	
Content Tables	
Footers	
Form Elements	
Headers	
Navigation	
Comments	
Comment, CSS/JS-Style, …	Turns the selection into a CSS/JS
Comment, CSS/JS-Style, S…	Turns the selection into a CSS/JS
Comment, Multi-Line	Wraps an HTML comment tag arc
Comment, Single-Line	Wraps an HTML comment tag arc
Development Note	Inserts an HTML comment into th
Content Tables	
Footers	
Title and Hanging List	An indented list of links with title
Two Columns	Copyright information in the left
Form Elements	
Headers	
JavaScript	
Meta	
Navigation	
Text	
~Legacy	

Figure 18-1:
The Snippets tab on the Files panel contains reusable chunks of code–snippets–which you can organize into folders. After selecting a snippet from the list, a preview appears in the Preview pane. In this example, you can see a preview of the snippet: a simple table with a colored background and dummy text. Snippets can have either a graphic preview (as in this example), called design preview, or a code preview, which is the raw code. Code previews are useful for snippets that aren't visible in Design view, such as JavaScript code. (When you create your own snippets, you specify the preview type.)

4. **In the Description field, type identifying details.**

   This step is optional, but useful. Use this field to provide a description of when and how to use the snippet and whether or not the snippet wraps a selection.

5. **Select a Snippet type.**

   "Wrap selection" makes the code wrap around a selection when you use the snippet in your Web pages. The "Insert block" option is for a snippet that's a single block of code inserted into the document—for example, a simple copyright notice, or a form menu.

6. **If necessary, add the code for the snippet.**

   If you initially selected code in the document window, it already appears in the "Insert before" field for a snippet that wraps around other code. For snippets that are just a single block of code, the code appears in the "Insert code" box.

If you're creating a wrapping snippet, then some code goes in the "before" field and some in the "after" field. For example, say you wanted to create a snippet that would let you set off a paragraph of text by adding a horizontal rule at the beginning of the paragraph and one at the end. In both the "Insert before" and "Insert after" fields, you'd type *<hr>*—the HTML code for a horizontal rule. (If you're creating XHTML pages as described on page 479, you'd type *<hr />* in both fields.)

7. **Select a "Preview type."**

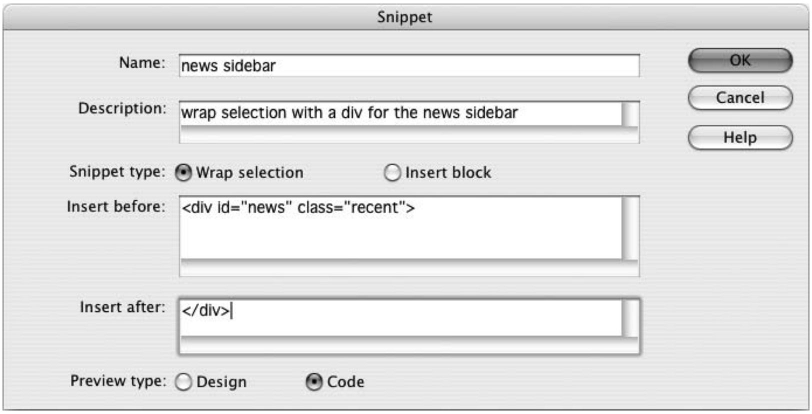
   The preview type determines how the snippet appears in the Preview pane of the Snippets tab (see Figure 18-1). *Design* means the snippet looks as it would in Design view—a snippet of a table appears as a table, for instance. *Code* means the code itself appears in the Preview pane (in that case, a snippet for a horizontal rule would preview like this: <hr>). Use Code preview for snippets that aren't visible in Design view, such as JavaScript code.

8. **Click OK.**

   Dreamweaver adds the snippet to the Snippets tab; you can then drop it in your Web pages using any of the techniques described on page 648.

If you need to go back and edit a snippet—change the code, type, description, or name—select the snippet in the Snippets tab and click Edit Snippet (Figure 18-1). You can also right-click (Control-click) the snippet name and then select Edit from the shortcut menu.

Whichever method you chose, the Snippet window (Figure 18-2) appears. Make your changes, and then click OK.



Name: news sidebar

Description: wrap selection with a div for the news sidebar

Snippet type: ● Wrap selection ○ Insert block

Insert before: <div id="news" class="recent">

Insert after: </div>

Preview type: ○ Design ● Code

Figure 18-2:
The Snippet window lets you create reusable chunks of HTML called snippets. For snippets that wrap around a currently selected object on the page–for example, a snippet that adds a link to any selected text or graphic–you put code in the two insert boxes. The code that appears before the selected object goes in the top box, and the code that goes after the object appears in the bottom box. In this example, the snippet wraps the current selection in a <div> tag (see page 313) with a predefined ID and class applied to it.

Organizing Snippets

To keep snippets organized, you can create new folders to store them by category. To add a folder to the Snippets tab, click the New Folder button (see Figure 18-1). An untitled folder appears; type a name for it. If you select a folder before clicking New Folder, Dreamweaver creates the new folder *inside* that folder. You can move folders around by dragging them into other folders.

Note: To drag a folder or snippet to the top level of the Snippets list, you have to drag it all the way to the *bottom* of the tab, below any other folders. If you try to drag it to the top, Dreamweaver puts the folder or snippet inside the top folder in the list.

To move a snippet into or out of its folder, simply drag it. If you drag a snippet over a closed folder without releasing the mouse, that folder expands to reveal the folders inside, if any.

To delete a snippet, select it from the Snippets tab and then click the Delete Snippet (Trash can) button (see Figure 18-1). Quicker yet, press Delete.

Note: Having lots of snippets can slow down the Snippets panel, so it's best to remove any snippets you don't use (like a lot of the ones that came with the program, for example). If you don't want to permanently delete them, you can move them out of the main Adobe Dreamweaver CS3 → configuration → Snippets folder and store them in a separate folder on your hard drive. (For more on the configuration folder and how to find it, see the box on page 744.)

Built-in Snippets

Many of the snippets that come with Dreamweaver offer solutions to specific problems you may never encounter, like a page footer containing two lists of links and a copyright notice. In addition, many use older design techniques (like using tables to lay out content) that are best avoided. However, most Web developers will find at least a few snippets worth using. Here are some highlights:

- **Close Window Button.** When you create a pop-up window (page 517), this snippet lets you add a Close button to let people dismiss the window. The "Close Window Button" snippet (in the Form Elements folder) places a form button with the words Close Window on the window page, complete with the JavaScript necessary to close the window when your visitor clicks the button.

- **Dropdown Menus.** If you create a lot of forms for your sites (see Chapter 11), you'll find some useful snippets in the Form Elements folder, especially in the Dropdown Menus subfolder. For example, the "Numbers 1-12" snippet inserts a menu with the numbers 1–12 already coded into it—great for capturing credit card expiration dates on an e-commerce site. (To create an even more useful drop-down snippet, see the tutorial on page 659.)

- **Autoclear Textfield.** Ever seen a Web form with a text field that has some helpful instructions *inside* the field ("Type your name here," for example)? This text magically disappears when you click in the field, leaving you free to type as instructed. A snippet called "Textfield, Autoclear" (also in the Form Elements folder) supplies both the text field and the bit of JavaScript that performs this little trick.

TROUBLESHOOTING MOMENT

A Snippet of Caution

Snippets aren't as smart as other Dreamweaver features. Dreamweaver is usually good about warning you before you make a mistake, but it doesn't make a peep if you're incorrectly adding a snippet.

For instance, when you use one of the program's form snippets to add, say, a text field to a page, Dreamweaver doesn't check to see if you're really putting the snippet into a form. Dreamweaver doesn't let you know if the required <form> tag is missing, and certainly doesn't add it itself. Furthermore, if you're working in Code view, Dreamweaver lets you add snippets to the <head> or even outside the <html> tags altogether, which is useful when creating dynamic Web pages that include server-side programming code, but just creates messy and invalid HTML on normal Web pages.

Furthermore, snippets don't take advantage of Dreamweaver's site management features to keep track of links or paths to images. Suppose you create a snippet that includes an image. If you insert that snippet into another page, the image may not show up correctly. If you create a snippet that includes a link from one page to another on your site, that link is also unlikely to work in another page.

So it's best to create snippets without images or links–but there are workarounds. For instance, you can create snippets with fake links–use nothing but the # symbol for the link, for example–and update the link after you insert the snippet into a page. For images, you can use Dreamweaver's Image Placeholder object to simulate a graphic in a snippet (choose Insert → Image Objects → Image Placeholder). After adding the snippet to the page, update the placeholder with your real image file.

If you want to create reusable content that can keep track of links and images, see Dreamweaver's Library feature, described below.

Library Basics

Imagine this situation: You manage a relatively large Web site consisting of thousands of Web pages. At the bottom of each page is a simple copyright notice: "Copyright MyBigCompany. We reserve all rights—national, international, commercial, noncommercial, and mineral—to the content contained on these pages."

Each time you add another page to the site, you *could* retype the copyright message, but this approach invites both typographic errors and carpal tunnel syndrome. And if you must *format* this text too, then you're in for even more work.

Fortunately, Dreamweaver's Library can turn any selection in the document window (a paragraph, an image, a table) into a reusable chunk of HTML that you can easily drop into any Dreamweaver document. The Library, in other words, is a great place to store copyright notices, navigation bars, or any other chunks of HTML you use frequently.

So far, this description sounds pretty much like the snippets described in the previous section. But Library items have added power: Each Library item that you add to a Web page is actually only a copy, which remains linked to the original. Thanks to this link, whenever you update the original Library item, you get a chance to update every page that uses that item.

Suppose your company is bought, for example, and the legal department orders you to change the copyright notice to "Copyright MyBigCompany, a subsidiary of aMuchBiggerCompany" on each of the Web site's *10,000 pages*. If you had cleverly inserted the original copyright notice as a Library item, you could take care of this task in the blink of an eye. Just open the item in the Library, make the required changes, save it, and let Dreamweaver update all the pages for you (see Figure 18-3).

Compared to Snippets, Library items are much smarter. They possess the unique ability to update the same material on an entire site's worth of files in seconds, and can successfully deal with links and images. Unlike Snippets, however, Dreamweaver's Library feature is site-specific. In other words, each site that you've defined in Dreamweaver has its own Library. You can't use a Library item from one site on a page from a different site.

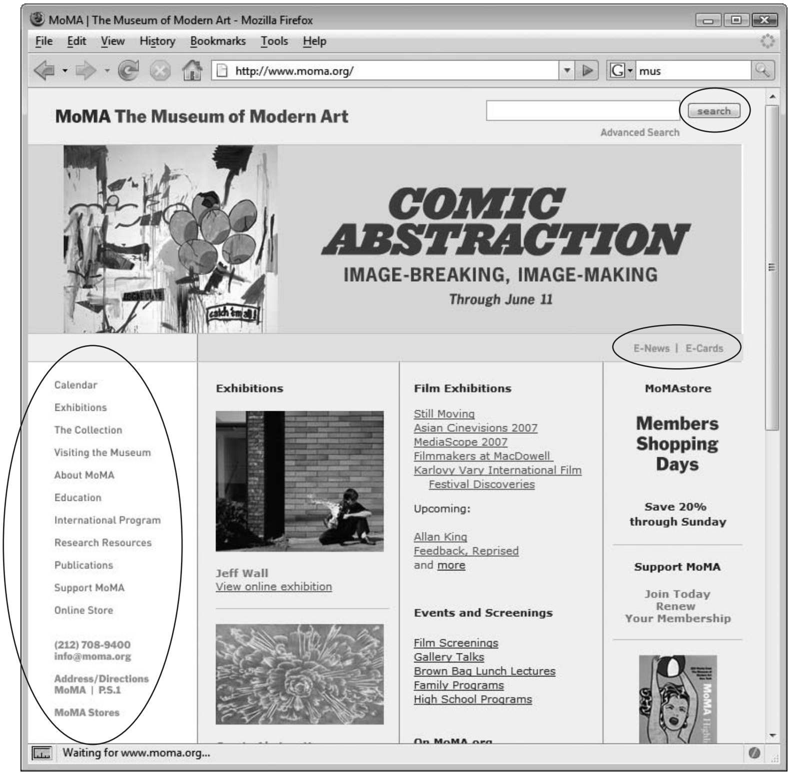


Figure 18-3: *The Museum of Modern Art's home page, which was created with Dreamweaver, takes advantage of Library elements. Many of the navigation options on the page (circled) are Library items. If the Museum decides to add or remove a navigation link, it can update the Library item to change every page on the site in one simple step (see page 657). In fact, since a Library item is a chunk of HTML, the Museum could decide to replace the left-hand navigation bar with a Flash movie, plain-text links (instead of graphics), or any other valid HTML code.*

Creating and Using Library Items

To create a Library item, start by opening the Library window. Choose Window → Assets, and click the Library Items button (it looks like an open book, circled in Figure 18-4) to reveal the Library category.

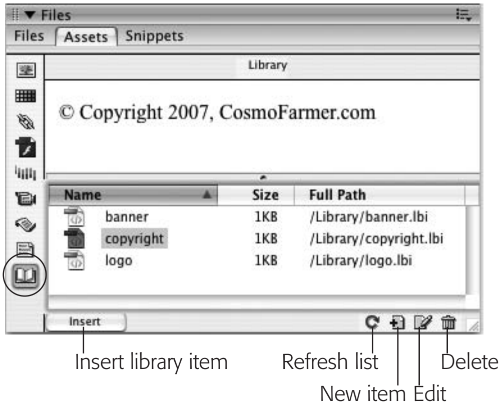
Now select the part of your document that you wish to save as a Library item: a blob of text, a graphic, or whatever.

Note, however, that Library items can contain only page elements that appear in the document window—in other words, only HTML from the <body> of a Web page. You can't include anything that appears in the <head> of a page, like Cascading Style Sheets, Dreamweaver Behaviors (Chapter 12), or meta tags. Furthermore, Library items must include a complete set of HTML tags—both an opening and closing tag—as well as all tags necessary to complete the original object. For example, Dreamweaver doesn't let you turn just a single cell, row, or column of a table into a Library item. If you try, Dreamweaver adds the *entire* table to the Library.

Tip: Use the Tag selector (see page 22) to make sure you select the precise tag information you want. To select *all* of the contents of a cell, click at the beginning of the content and drag until you've selected everything in the cell.

Next, add the selection to the Library. As you may expect, Dreamweaver provides several ways to do this:

- Drag the highlighted selection into the list of Library items.
- Click the New Item button (Figure 18-4).
- Choose Modify → Library → "Add Object to Library".



© Copyright 2007, CosmoFarmer.com

Name	Size	Full Path
banner	1KB	/Library/banner.lbi
copyright	1KB	/Library/copyright.lbi
logo	1KB	/Library/logo.lbi

Figure 18-4:
The Assets panel's Library category lists the name, file size, and location of each Library item in the currently opened site. When you select a Library item from the list, you see a small preview. In this example, the Library item "copyright" is a copyright notice.

The new item appears in the Assets panel, bearing the jaunty name "Untitled." Just type to replace that with a more useful name, such as *Copyright notice* or *Logo*.

(Avoid hyphens in your Library item's name. These tend to trip up the Firefox Web browser, as described in the box on page 659.) Your new Library element is ready to use.

Note: Even though you can't turn a CSS style into a Library item, you *can* turn HTML that has been styled with CSS into a Library item. For example, you can add to the Library a paragraph that has a CSS class style applied to it. When you attempt to add this paragraph to the Library, Dreamweaver warns you that the item may not look the same when you place it in other documents–because the style sheet information doesn't come along for the ride. To make sure the Library item appears correctly, make sure that you attach the same style sheet to any page where you use that item. External style sheets (see page 113) make this easy.

Adding Library Items to a Page

To add a Library item to a Web page, drag it directly out of the Assets panel's Library items listing onto your page. (The long way: Click to plant your insertion point in the Web page, click the Library item you want in the Assets panel, and then click the Insert button on the Assets panel, shown in Figure 18-4.)

Note: Library items (.lbi files) also appear in the Files panel in a site's Library folder. Dragging a Library item from the Files panel to a page, however, *doesn't* insert it into the page. It adds the name of the library item file (not its contents) with a link to the .lbi file–not something you want to do.

When you insert a Library item into a Web page (or turn a selected item *into* a Library item), it sprouts a light-yellow background color. The highlighting indicates that Dreamweaver intends to treat the Library item as a single object, even though it may be made of many different HTML elements. You can select it or drag it around, but you can't change it on the Web page. (Unfortunately, if you turn a nontransparent graphic into a Library item—like a logo, for example—Dreamweaver doesn't give you this helpful visual cue.)

Remember, too, that the placed Library item is linked to the original copy in the Library. The copy in your document automatically changes to reflect any changes you make to the copy in the Library, using the technique described next.

Tip: Sometimes you may want to sever the connection between the Library and a Library item you've already placed onto a Web page–to modify a copyright notice on a particular page, for example. Select the item on the page and then click "Detach from original" in the Property inspector (Figure 18-5). Dreamweaver removes the comment tags (see the box on page 659), thus breaking the link to the Library.

You can also insert the HTML of a Library item *without* maintaining a link to the Library by pressing the Ctrl (⌘) key when you add it to your document. Now Dreamweaver doesn't update the HTML on this page when you change the original Library file.

Don't use this method if the Library item contains images or document-relative links, however. Dreamweaver doesn't update the links with paths appropriate to the document's location when you insert the Library item in this way. In this case, first insert the Library item normally and then unlink it using the method described in this tip's first paragraph.

Editing Library Items

You'll appreciate the real power of Library items when it's time to make a change. When you update the file in the Library, all pages that you've graced with that item update themselves, too.

Start by opening the Library, as described on page 655. Then:

1. **Open the Library item that you want to edit.**

   You can do this by double-clicking the Library item in the Assets panel, by highlighting it and then clicking the Edit button (Figure 18-4), or by highlighting a Library item on a Web page and then clicking the Open button on the Property inspector (Figure 18-5). (You can also open the Library item file—an .lbi file—in the Library folder of your site's root directly from the Files panel.)

   Dreamweaver opens what looks like a normal Web page document, but it only contains the text, graphics, or other elements of the Library file.

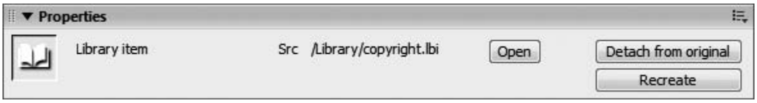


Figure 18-5: *The selected Library item (a .lbi file) is in the site's Library folder. (The path appears after the word "Src.")*

2. **Edit away.**

   A Library item is only a selection of HTML; it's not a complete Web page. That means you shouldn't add *page* properties like the title or background color (Dreamweaver will actually let you do this, but that will add invalid HTML code to the Library item as well as every page that uses that Library item). Also, you can insert Library items only in the body of a Web page, so stick with objects that would normally appear in the document window, such as links, images, tables, and text. Don't add any code that appears in the head of a Web page, such as Cascading Style Sheets, meta tags, behaviors, or timelines.

   And since a Library item can't contain a style sheet, if the HTML in your Library item relies on a style, you'll have trouble previewing it correctly. Dreamweaver's Design Time Style Sheet tool comes in handy here. It lets you temporarily "add" a style sheet while designing a page, without actually adding the CSS code to the page. For more on this cool feature, turn to page 307.

Note: Don't turn any of Dreamweaver CS3's Spry widgets into Library items. For example, if you use the Spry Menu Bar (page 175) you might be tempted to turn the menu into a Library item that you could reuse on other pages of your site. Problem is, all Spry features combine HTML, JavaScript, and CSS code that are placed in different parts of a page's code. When you select the Spry widget on the page and turn it into a Library item, only the HTML comes along for the ride–the CSS which makes the widget look good, and the JavaScript which make the widget work, aren't included. The solution? Use Dreamweaver templates instead (see the next chapter).

3. **Choose File → Save.**

   Dreamweaver checks to see if there are any pages that use the Library item, and, if there are, it opens the Update Library Items window. A list of pages in the site that use that Library item appears.

4. **Click Update.**

   Dreamweaver opens the Update Pages window, updates the HTML in all the pages that use the Library item, and then lists all of the files that it changed.

   On the other hand, you don't necessarily have to click Update. Perhaps you have a lot of changes to make to the Library item, and you just want to save the work you've done so far. You're not done editing it yet, so you don't want to waste time updating pages you'll just have to update again. You can always update later (see the box on page 699); in that case, click Don't Update. (Once you're finished with the changes and save the file for the final time, *then* update the site.)

5. **Click Done.**

   As you can see, the Library is an incredible time-saver that greatly simplifies the process of changing common page elements.

Renaming Library Elements

To rename something in your Library, click its name on the Assets panel (Figure 18-4). Pause briefly, then click again, and the name highlights, ready to be edited. Type the new name and press Enter (Return).

If you've already added the item to your Web pages, Dreamweaver prompts you to update those pages. Click Update. Otherwise, the link between those pages and the Library breaks.

Note: If you accidentally click Don't Update, don't panic. Simply change the Library item back to its original name and then *re*-rename it. Don't forget to click Update this time!

Deleting Library Elements

You can delete unnecessary elements from your Library at any time, but do so with caution. When you delete something from the Library, Dreamweaver leaves behind every copy of it that you've already placed onto your Web pages—complete with links to the now-deleted Library item.

In other words, you won't be able to edit the copies on your Web pages until you break those links. If you do indeed want to edit them, you have to break the links manually on each page where the Library item appears by selecting the item and then clicking the "Detach from original" button (see Figure 18-5).

POWER USERS' CLINIC

Under the Hood of Library Items

Behind the scenes, Dreamweaver stores the HTML for Library items in basic text files. Those files' names end with the extension .lbi, and they stay in the Library folder inside your local site folder.

When you insert a Library item into a Web page, Dreamweaver inserts the item's HTML and adds a set of comment tags. These tags refer to the original Library file and help Dreamweaver remember where the Library item begins and ends. For instance, if you turned the text "Copyright 2007" into a Library item called *copyright* and inserted it into a Web page, Dreamweaver would add the following HTML to the page:

```
<!-- #BeginLibraryItem "/Library/
copyright. lbi" -->Copyright 2007<!--
#EndLibraryItem-->
```

It's important to avoid the use of hyphens when naming Library items. Why? Since HTML comments use hyphens, <!-- -->, some browsers, most notably Firefox, get tripped up by additional hyphens and respond by hiding the contents of a Library item, or displaying raw HTML code instead.

In addition, although you can't edit a Library item on a page in Design view, you can muck around with the code in Code view. In the example above, you could change 2007 to 2008 in Code view. Don't do it! Dreamweaver obliterates any changes you make the next time you update the original Library item. If you want to make a change to a Library item, edit the original Library item, or detach the item from the Library (as described in the Tip on page 656) and then edit it.

Now that you've been warned, here are the instructions. To get rid of a Library item, click it in the Assets panel and then do one of the following:

- Click the Trash can icon in the Assets panel.
- Press Delete.
- Right-click (Control-click) the item's name, and then choose Delete from the shortcut menu.

Tip: If you ever accidentally delete an item from the Library, you can recreate it, provided you've used it somewhere on one of the Web pages in the site.

Open the page containing the Library item, and then click the Library item to select it. Click Recreate on the Property inspector (Figure 18-5) to make it anew. A new Library item appears in the Library, using the name and HTML from the item you selected.

Snippets and Library Tutorial

In this tutorial, you'll do two things: First, create a useful form pull-down menu snippet, and, second, turn the *CosmoFarmer* copyright notice into a reusable Library item and add it to several pages in the site.

Note: You'll need to download the tutorial files from *www.sawmac.com/dwcs3/* to complete this tutorial. See the Note on page 39 for more details.

Once you've downloaded the tutorial files and opened Dreamweaver, define a new site as described on page 28: name the site *Snippets and Library* and select the Chapter18 folder (inside the MM_DWCS3 folder). (In a nutshell: choose Site → New Site. In the Site Definition window, click the Advanced tab, type *Snippets and Library* into the Site Name field, click the folder icon next to the Local Root Folder field, navigate to and select the Chapter18 folder, and then click Choose or Select. Finally, click OK.)

Creating a Snippet

1. **With your site freshly defined, make sure the Files panel is open.**

   If the panel isn't open, press the F8 key or choose Window → Files.

2. **In the Files panel, double-click the file, *snippet.html.***

   A page with several form pull-down menus opens. The page includes menus for the months of the year, names of U.S. states, and the numbers 1–31. These menus are useful for specifying dates when something needs to be done, states for shipping orders to, or simply for selecting a month for one's astrological sign. Dreamweaver's own Snippets don't include these useful menus, but, fortunately, you can add them yourself.

3. **Click the first form menu at the top of the page.**

   This menu appears to the right of the words "Months of the Year." You've selected the menu (and its underlying HTML code). To add this as a snippet you need to open the Snippets panel.

4. **Click the Snippets panel tab to the right of the Files panel, or choose Window → Snippets.**

   The Snippets panel (Figure 18-1) is your control center for adding, editing, and deleting Snippets.

5. **Click the New Snippet button at the bottom of the panel (Figure 18-1).**

   The Snippet window opens. Dreamweaver automatically copies the code for the menu into the window. You just need to name the snippet and add a few more details.

6. **Type *Month Menu* in the Name box, and *A list of month names, with numeric values* in the description box, as pictured in Figure 18-6.**

   The name and description appear in the Snippets panel. In this case, the description identifies what appears in the menu on the page (a list of month names) and what value someone submits when selecting a month from the list and submitting the form—in other words, the name/value pair for this form field. (See Chapter 11 for more information on how forms work.)

7. **Select the "Insert block" radio button.**

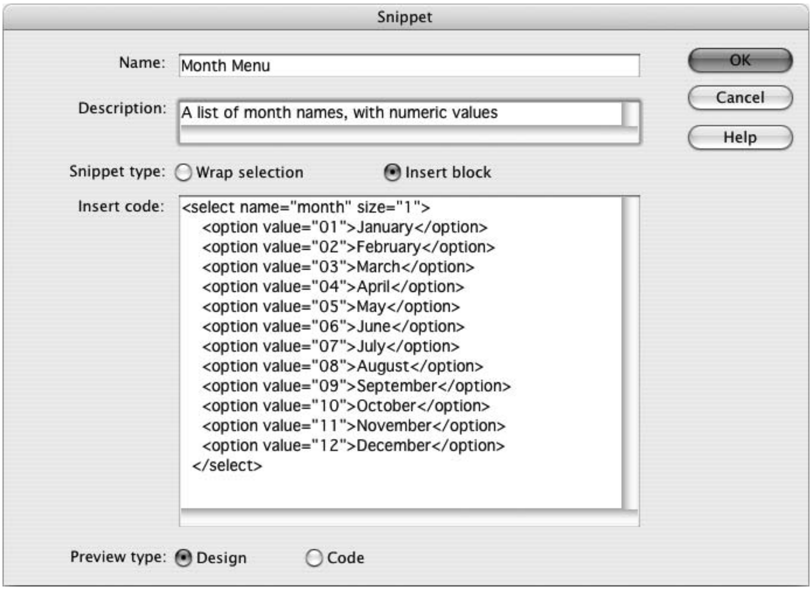
   This button identifies the snippet as a chunk of HTML that's simply plopped down on a page, as opposed to HTML that wraps around a selected graphic or text like a link or table cell might (if you wanted to do that, you'd select the "Wrap selection" button).

8. **Select the Design button at the bottom of the window.**

   You've just told Dreamweaver to display the snippet visually when it's selected in the Snippets panel. In other words, when you select this snippet in the panel, you see a preview of the form menu, not a bunch of HTML code.

9. **The window should now look like Figure 18-6. Click the OK button to create your new snippet.**

   The snippet should now appear in the Snippets panel, ready to be inserted into a page.



Name: Month Menu

Description: A list of month names, with numeric values

Snippet type: ○ Wrap selection ● Insert block

Insert code:

```
<select name="month" size="1">
   <option value="01">January</option>
   <option value="02">February</option>
   <option value="03">March</option>
   <option value="04">April</option>
   <option value="05">May</option>
   <option value="06">June</option>
   <option value="07">July</option>
   <option value="08">August</option>
   <option value="09">September</option>
   <option value="10">October</option>
   <option value="11">November</option>
   <option value="12">December</option>
 </select>
```

Preview type: ● Design ○ Code

Figure 18-6: *The HTML code for the snippet appears in the "Insert code" box. You can edit it, adding or removing HTML, or you can create code that wraps around anything selected on the page by choosing the "Wrap selection" button.*

10. **Select the Files panel by clicking the Files tab or pressing the F8 key; then double-click the file *form.html*.**

    This opens a CosmoFarmer page. You'll insert the new snippet below the headline "What month were you born?" But first you need to create the form. Remember that snippets don't have smarts; that is, when you insert a snippet, Dreamweaver doesn't make sure the HTML is correct. In this case, the snippet is just a menu—for it to really function, it has to be part of a form, so you first need to insert a form field.

11. **Choose Insert → Form → Form.**

    A dashed red line appears on the page, indicating the outline of the form. Inside this area you can insert any form element you want, including your new snippet.

12. **Return to the Snippet panel once again by clicking the Snippets tab or choosing Window → Snippets.**

    Now for the moment of truth.

13. **Drag your new snippet—Month Menu—from the Snippets panel to anywhere inside the form's red outline.**

    Ta-da, Dreamweaver adds the new menu. Now, whenever you need to add a menu listing the months of the year, don't bother creating it from scratch. Just use the snippet! If you wish, you can create more snippets using the other menus on the *snippet.html* page.

Creating a Library Item

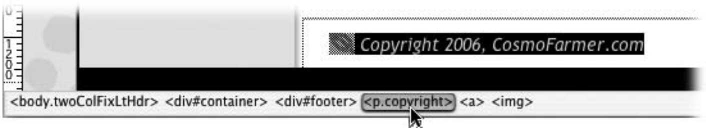
Now you'll see one way in which Dreamweaver's powerful Site Management tools can help you create and update your Web sites more effectively:

1. **In the Files panel, double-click the file *hydroponics.html*.**

   This is the same page you created in the tutorial for Chapter 9. What you're going to work on here is the copyright notice in the page's footer.

2. **Scroll to the bottom of the page and select the copyright notice.**

   Copyright information should be on every page of the site, so it makes a perfect candidate for a Library item. Notice that the copyright information is more than just text; it's actually a paragraph including a small image that's linked to the home page. The best way to select the entire paragraph is to click inside the text and then click the <p.copyright> tag in the Tag selector (Figure 18-7).



Copyright 2006, CosmoFarmer.com

<body.twoColFixLtHdr> <div#container> <div#footer> <p.copyright> <a> <img>

Figure 18-7:
Click <p.copyright> to select the paragraph containing the site's copyright notice.

3. **Choose Window → Assets, and then click the Library button.**

   The Assets panel opens and displays the Library category.

4. **Click the New Library Item button on the Assets panel (Figure 18-4).**

   A warning message appears, saying that the Library item may not look the same in other pages. Dreamweaver's trying to tell you that Library items can contain only HTML from the body of a Web page—not Cascading Style Sheets. (You can still include HTML, such as this paragraph, that's had a style applied to it, as long as you make sure that any *pages* to which you add the Library item have the appropriate style sheets.)

The text in this example *is* formatted using a style sheet, so, sure enough, it won't look the same in pages that don't have the same style sheet. In this exercise, however, this formatting isn't a problem, since all the pages in the site share the same linked external style sheet (see page 113).

Click OK to dismiss the warning. The copyright notice item appears in the Library list, with an "Untitled" naming rectangle next to it.

5. **Type *copyright* next to the new item on the Assets panel, and then press Enter.**

You've just checked this standard blob of text into your Library. It's ready to use anywhere else on your site.

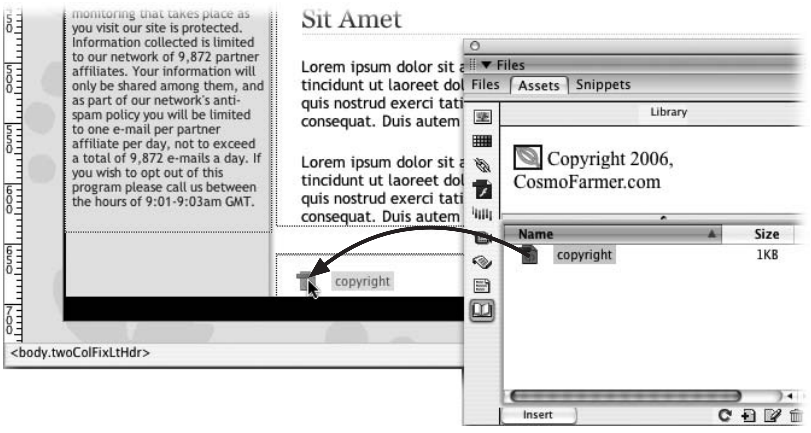
6. **In the Files panel, double-click the file called *index.html*.**

Notice that this page is missing a copyright notice at the bottom of the page.

You'll frequently jump between the Files panel and the Assets panel, so these keyboard shortcuts come in handy: the F8 key to open the Files panel, and the F11 key (Option-F11 on Macs) to switch to the Assets panel.

7. **Switch back to the Assets panel, and drag the copyright Library item to the bottom of the page, as shown in Figure 18-8.**

You can recognize the newly inserted Library item by its yellow background. Click the text in the item and notice that you can't edit it; Dreamweaver treats it like a single object. (You may notice an extra space below the copyright you just added. That's an empty paragraph that was already in the footer. If you click in the empty space directly below the copyright and press the delete key, you'll remove the empty paragraph.)



monitoring that takes place as you visit our site is protected. Information collected is limited to our network of 9,872 partner affiliates. Your information will only be shared among them, and as part of our network's anti-spam policy you will be limited to one e-mail per partner affiliate per day, not to exceed a total of 9,872 e-mails a day. If you wish to opt out of this program please call us between the hours of 9:01-9:03am GMT.

Sit Amet

Lorem ipsum dolor sit a tincidunt ut laoreet dol quis nostrud exerci tati consequat. Duis autem

Lorem ipsum dolor sit a tincidunt ut laoreet dol quis nostrud exerci tati consequat. Duis autem

Copyright 2006, CosmoFarmer.com

***Figure 18-8:** In addition to dragging a Library item into the document window, you can also insert the item by placing the insertion point in the document window and then clicking the Insert button on the Assets panel.*

8. **Add the footer Library item to one other page on the site: *feature1.html*.**

   Open the page (by double-clicking its name in the Site window) and repeat step 7.

   (You can close and save the pages as you go, or leave them open. Leave at least one open at the end and go on to step 9.)

9. **Notice the mistake!**

   The copyright notice has the wrong date. The year 2006 is long gone! Oh, great—you'll have to change the date on every page. Fortunately, you've used a Library item, so you can easily make the change.

10. **Double-click the copyright item's icon** (**not its name**) **in the Assets panel.**

    The Library item opens up, ready for editing.

11. **Change 2006 to the current year. Choose File → Save.**

    The Update Library Items dialog box appears, listing all of the pages in the site that use the footer item.

12. **Click Update.**

    Dreamweaver opens the Update Pages dialog box and updates all the Web pages that use the footer item.

13. **Click Close to close the Update Pages dialog box.**

    And *now* if you open *index.html*, *feature1.html*, and *hydroponics.html*, you'll find that the copyright date is correct on all three.

    Now imagine that you just used this auto-update feature on a 10,000-page site. Sit back and smile.

CHAPTER
19

Templates

Some Web designers handcraft sites with loving care, changing layouts, colors, fonts, banners, and navigation from page to page. But that approach isn't always practical—or desirable. Consistency is a good thing. Web pages that look and act similarly reassure visitors; when only important material changes from page to page, readers can concentrate on finding the information they want. Even more importantly, a handcrafted approach is often unrealistic when you're designing on a deadline.

Here's where *templates* come in. Frequently, the underlying design of many pages on many Web sites is identical (see Figure 19-1). For instance, a company Web site with an employee directory may dedicate a single Web page to each employee. Each employee page probably has the same navigation bar, banner, footer, and layout. Only a few particulars differ, like the employee name, photo, and contact information.

Template Basics

Templates let you build pages that share a similar structure and graphic identity, quickly and without having to worry about accidentally deleting or changing elements. Templates come in very handy when you're designing a site for which other, less Dreamweaver-savvy individuals are responsible for adding new pages. If you use a template, these underlings can modify only the areas of a page that you, the godlike Dreamweaver guru, define.

Tip: Adobe Contribute, a simple, word processor–like program for updating Web sites, works very well with sites built using Dreamweaver templates. If you build sites that are updated by people who don't know the first thing about Dreamweaver or building Web pages, Contribute can help. You can find more information about this program at Adobe's Web site: *www.adobe.com/products/contribute*.

A new page based on a template—also called a template *instance*, or *child page*—looks just like the template, except that you can edit only certain areas of the page, called, logically enough, *editable regions*. In the example shown in Figure 19-1, one editable region includes the question-and-answer text area; the rest of the page remains untouched and is, in fact, locked.

A Dreamweaver template can be very basic: one or more areas of a page (the editable regions) can be changed, others can't (*locked regions*). But Dreamweaver also includes many subtle ways for controlling template instances. Here's an overview of the features you'll encounter when creating and using templates:

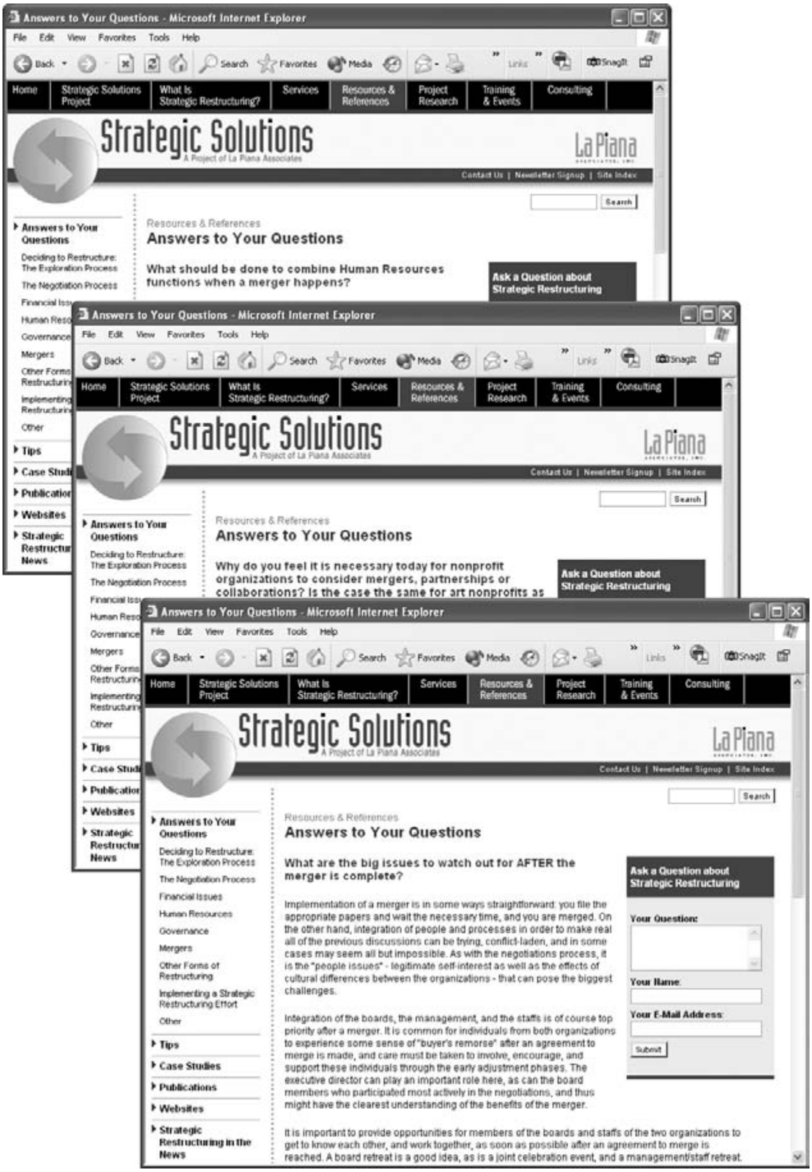
- **Editable regions.** These are the basic building blocks of a template. An editable region is a part of a page—a paragraph, the contents of a div, or a headline, for example—that people can change on each template instance. A template page can have multiple editable regions—for example, one in a sidebar area and another in the main content section of a page.

- **Editable tag attributes.** There may be times when you want to make a particular *tag* property editable. For instance, you might want to assign an ID to the <body> tag. Then, on each template-based page, you could apply an ID specific to a particular section of the site. For a template-based page that you save as part of an "About Us" section of the site, you could apply an ID with the name *about*. With that in place, you could use a descendent selector (like *#about h1)* to create a custom style for all headlines on that page. You'll see this technique in the tutorial on page 705.

  Or perhaps you've built a template that includes a photo with some complex formatting (left-aligned by a Cascading Style Sheet, perhaps). Turning the entire image into an editable region could pose problems: When someone creates a new page from the template and then inserts a new photo, all of the formatting information can get lost. Instead, you could make just the image's *Src* property editable. People would then be able to insert new images for each page without inadvertently ruining the photo's formatting. (You could also make the image's *alt* property editable and, if the *Width* and *Height* properties vary from image to image, you could make those attributes editable as well.)

- **Repeating regions and repeating tables.** Some Web pages include *lists* of items: catalogs of products, lists of news articles, albums of photos, and so on. Dreamweaver lets you define *repeatable* regions for pages like this.

  For example, a page of product listings can include a picture, name, and price for each product in a catalog, organized using a table with multiple rows (Chapter 7).



Answers to Your Questions - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media Links SnagIt
Home
Strategic Solutions Project
What Is Strategic Restructuring?
Services
Resources & References
Project Research
Training & Events
Consulting
Strategic Solutions
A Project of La Piana Associates
La Piana
Contact Us | Newsletter Signup | Site Index
Search
Answers to Your Questions
Deciding to Restructure: The Exploration Process
The Negotiation Process
Financial Issues
Human Resources
Governance
Mergers
Other Forms of Restructuring
Implementing a Strategic Restructuring Effort
Other
Tips
Case Studies
Publications
Websites
Strategic Restructuring in the News
Ask a Question about Strategic Restructuring
Your Question:
Your Name:
Your E-Mail Address:
Submit

Resources & References

Answers to Your Questions

What should be done to combine Human Resources functions when a merger happens?

Resources & References

Answers to Your Questions

Why do you feel it is necessary today for nonprofit organizations to consider mergers, partnerships or collaborations? Is the case the same for art nonprofits as

Resources & References

Answers to Your Questions

What are the big issues to watch out for AFTER the merger is complete?

Implementation of a merger is in some ways straightforward: you file the appropriate papers and wait the necessary time, and you are merged. On the other hand, integration of people and processes in order to make real all of the previous discussions can be trying, conflict-laden, and in some cases may seem all but impossible. As with the negotiations process, it is the "people issues" - legitimate self-interest as well as the effects of cultural differences between the organizations - that can pose the biggest challenges.

Integration of the boards, the management, and the staffs is of course top priority after a merger. It is common for individuals from both organizations to experience some sense of "buyer's remorse" after an agreement to merge is made, and care must be taken to involve, encourage, and support these individuals through the early adjustment phases. The executive director can play an important role here, as can the board members who participated most actively in the negotiations, and thus might have the clearest understanding of the benefits of the merger.

It is important to provide opportunities for members of the boards and staffs of the two organizations to get to know each other, and work together, as soon as possible after an agreement to merge is reached. A board retreat is a good idea, as is a joint celebration event, and a management/staff retreat

Figure 19-1:
These three Web pages are part of a section of a Web site dedicated to answering frequently asked questions. The pages each provide the answer to a different question, but are otherwise identical, sharing the same banner, navigation buttons, sidebar, and footer. This is a common scenario for Web sites that include news stories, employee profiles, product pages, or press releases. In fact, it's so common that Dreamweaver has a special feature–Templates–to help you build such pages.

As the template builder, you may not know in advance how many products the page will eventually list, so you can't fully design the page. However, you can use Dreamweaver to define a row—or any selection of HTML—as a repeating region, so that page authors can add new rows of product information when needed.

- **Optional regions and editable optional regions.** Optional regions make templates even more flexible. They let you show or hide content on a page-by-page basis.

> Suppose you create a template for your company's products. When some products go on sale (but others remain full price), you could add an *optional* region on the template that displays a big "On Sale!" logo. When you create a new product page, you could *show* the optional region for products that are on sale and keep it *hidden* for the others.
>
> Editable optional regions are similar, but they have the added benefit of being editable. Maybe you're creating a template for an employee directory, giving each employee a separate Web page with contact information. Some employees also want their picture displayed on the page, while others don't (you know the type). Solution: Add an editable optional region that would let you show the space for a photo and add a different photo for each page. For the shyer types, you would simply hide the photo area entirely.

Furthermore, Dreamweaver can create *nested* templates, which inherit design elements from a master template. In this way, you can create a general unified design that's shared by other templates; this feature is described on page 687.

But facilitating page creation is only one of the benefits of templates. You'll also find that templates can greatly simplify the process of updating a Web site's design. Like Library items, pages based on templates retain a reference to the original template file. Any changes made to the template pass on to all pages created from it, which can save you hours of time and trouble when it comes time to update the look or structure of your site. Imagine how much time you'll save when your boss asks you to add "just one more" button to the site's navigation bar. Instead of updating thousands of pages by hand, you need to update only a single template file.

Creating a Template

The first step in creating a template requires building a basic Web page and telling Dreamweaver you'd like to use it as a template. You can go about this in two ways: build a Web page and turn it into a template, or create a blank, empty template file and add text, graphics, tables, and other content to it.

Turning a Web Page into a Template

The easiest way to create a template is simply to base it on a Web page in your current site folder. Although you can create templates based on Web pages that *aren't* part of the current local site, you may run into problems with links and paths to images, as described in a moment.

Once you've opened the Web page, just choose File → Save As Template or, on the Common tab of the Insert bar (see Figure 19-2), click the Templates button and then select Make Template from the menu. In the Save As Template window (Figure 19-3), the name of the current local site appears in the Site pop-up menu; meanwhile, all templates for that site show up in the Existing Templates field.

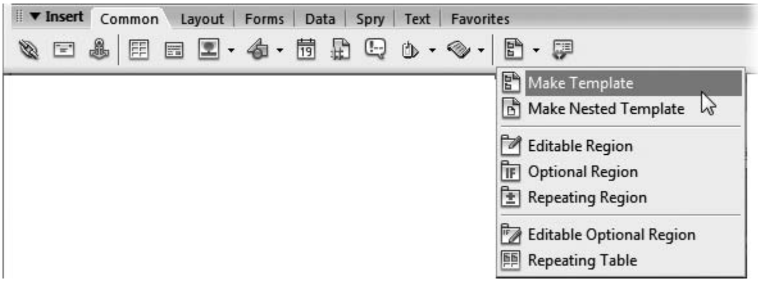
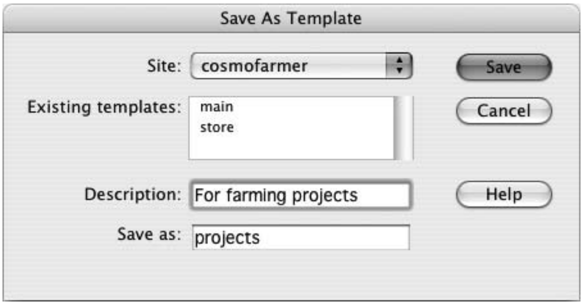


Figure 19-2:
The Templates menu on the Common tab of the Insert bar provides access to tools for creating templates and setting up a variety of Dreamweaver template features.


Save As Template

Site: cosmofarmer

Existing templates: main, store

Description: For farming projects

Save as: projects

Figure 19-3:
The Save As Template dialog box lets you save your template into any of the local site folders you've defined in Dreamweaver. Stick to your current local site to avoid broken links and similar problems.

Note: At this point, you could theoretically use the Site menu to save a template into any local site folder you've defined (see Chapter 15 for a discussion of local sites), but be careful with this option. If your page contains images and links and you save it as a template for another local site, Dreamweaver doesn't copy the images from the first site folder into the other one. As a result, the paths to the image files and links don't work correctly.

If you must use a page from one site as a template for another, copy the Web page *and graphics* into the new site's root folder, open the page from there, and then create a template as described here.

Dreamweaver includes a Description field for adding a brief note describing the template. This description appears when you're selecting a template as the basis for a new page you're creating. The description is very useful when *other* people are building a site using your templates and aren't sure whether templateA1, templateA2, or templateA3 is the correct choice; a simple "use this template for all FAQ pages" is much clearer.

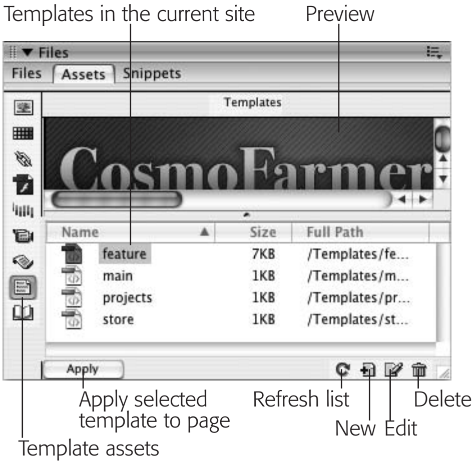
Finally, type a name for the new template, and then click Save. Choose Yes when Dreamweaver asks if you want to Update Links for the page. If you choose No, all page-relative links break, and all the images on the page appear as broken-image icons.

Warning: Avoid double hyphens in the names of your templates–for example, "Store -- Product." Since Dreamweaver uses HTML comments like this <!-- --> (see the box on page 674) to identify what template was used on a particular page, the extra hyphens trip up the Firefox Web browser, which mistakes them for other HTML comments.

Dreamweaver saves the page in the Templates folder of your local site root folder. It adds the extension .dwt to the file to indicate that it's a Dreamweaver template. (For dynamic Web pages, Dreamweaver adds the .dwt *before* the file's extension. For example, a PHP template may have a name like *maintemplate.dwt.php*.)

Building a Template from Scratch

It's easiest to create a Web page first and then save it as a template, but you can also build one from scratch. Open the Asset panel's Templates category by choosing Window → Assets and then click the Template assets icon (see Figure 19-4). Then click the New Template button at the bottom of the Assets panel. Once Dreamweaver adds a new, untitled template to the list, type a new name for it. Something descriptive like "press release" or "employee page" helps you keep track of your templates.



Name	Size	Full Path
feature	7KB	/Templates/fe...
main	1KB	/Templates/m...
projects	1KB	/Templates/pr...
store	1KB	/Templates/st...

Figure 19-4:
The Templates category of the Assets panel lists the name, file size, and location of each template in the current local site. The Apply button applies a template to the current open Web page. The Refresh Site List button updates the list of templates: If you've just created a template and don't see it listed, click this button. The New Template button creates a new blank template in the Templates folder. Select a template from the list and click the Edit Template button to open the template for editing.

After you've created a blank template for the site, you can open it by double-clicking its name in the Assets panel (or selecting its name and then clicking the Edit button at the bottom of the Assets panel). It opens just like any Web page, so that you can get busy designing it.

Defining Editable Regions

Your next task is to specify which parts of your template are locked and which are editable. By default, *everything* on a page is locked. After all, the main reason to use templates is to maintain a consistent, unchanging design and structure among pages. To make a template usable, you must define the area or areas you *can* change.

Adding a Basic Editable Region

To add an editable region to a template, start by selecting the part of the page you want to make changeable. You can designate as editable anything in the document window (that is, any HTML between the <body> tags).

Note: You can always add Cascading Style Sheets, JavaScript code, and meta tag information to the <head> of a template-based page. Any <head> content in the original template files stays put, however. For example, you can't remove an external style sheet applied to the template file from a page based on that template.

FREQUENTLY ASKED QUESTION

The Broken-Link Blues

Why aren't the links in my templates working?

When you created the link, you probably typed a path into the Property inspector's Link field–a recipe for heartbreak. Instead, always select the target Web page for a link by clicking the folder icon in the Property inspector, or by pressing Ctrl+L (⌘-L). In other words, when adding links to a template, always link to pages within the site by browsing to the desired file.

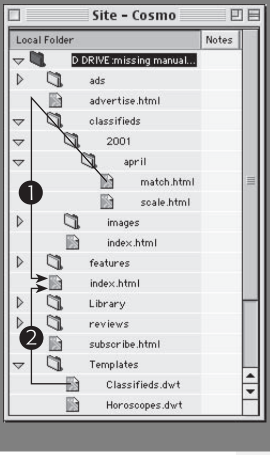
Dreamweaver saves templates in the Templates folder inside the local root folder; all relative links need to be relative to this location. (Absolute links, like those to other Web sites, aren't a problem; see page 153 to learn the difference.) The reason you should browse to, rather than type in, your links is so that Dreamweaver can create a proper relative link.

Imagine this situation: You create a template for your classified ads. You store all classified ads for April 2001 inside a series of folders like this: classifieds → 2001 → april, as shown in the site diagram here.

A link from a page in the *april* folder to the home page would follow the path marked 1 here. So when you create a link in your template, you can create a link to the home page by typing the path *../../../index.html*.

That choice is logical if you're thinking about the page (in the *april* folder) you'll create from the template–but it won't work. Dreamweaver stores templates in the Templates folder, so the correct path would be path 2, or *../index.html*. When you create a new page based on the template and save it in the *april* folder, Dreamweaver, in its wisdom, automatically rewrites all paths in the page so that the links function correctly.

The beauty of Dreamweaver is that you don't have to understand how all this works. Just remember to use relative links in your templates and create them by clicking the folder icon in the Property inspector.



Drag across your page to select the elements you wish to make editable, or, for greater precision, use the Tag selector (see page 22) to make sure you select the exact HTML you want.

Now tell Dreamweaver that you want to make the selected elements editable. You can use any of these techniques:

- In the Common tab of the Insert bar (Figure 19-2), select Editable Region from the Template menu.
- Choose Insert → Template Objects → Editable Region.
- Press Ctrl+Alt+V (⌘-Option-V).
- Right-click (Control-click) the selection and then choose Templates → New Editable Region from the shortcut menu.

FREQUENTLY ASKED QUESTION

When Save Won't Behave

I keep getting an error message when I save my template. What's going on?

If you add an editable region *inside* certain block-level elements like a paragraph, or a heading, Dreamweaver pops up a warning message when you save the template, explaining that you can't create additional paragraphs or headings inside this region on any pages you build from this template. This just means that you didn't select the <p> or heading tag when you made the region editable. Dreamweaver considers anything outside of the editable region locked, so you can't change those tags. Since it's improper HTML to have a paragraph, heading, or other block-level elements inside *another* paragraph, or heading, Dreamweaver doesn't let you add other block-level elements to the selection.

This characteristic may not be such a bad thing, however. Imagine you're creating a template that's to be used by other people building a Web site. You have a heading 1, maybe the title of the page you applied a style to, and you want to make sure it looks the same on every page. You wouldn't want anyone changing the heading tag, and possibly erasing the style. In addition, you don't want them to be able to change the heading 1 to a heading 2, or a heading 3; nor do you want them to completely erase the h1 tag and type paragraph after paragraph of their random thoughts. You just want them to type in new text for the page title. Selecting just the text inside the heading and turning it into an editable region does just that. Viva micro-management!

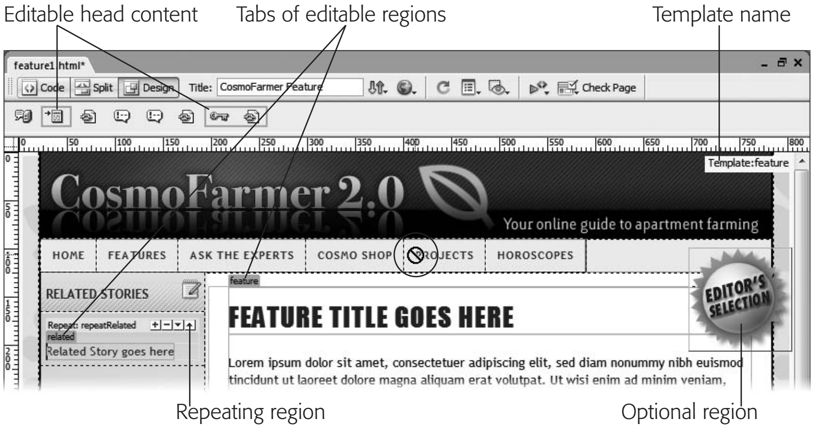
If this is in fact what you want to do, you can save yourself the bother of having to constantly see the warning box shown here each time you save the template by simply turning on the "Don't show me this message again" checkbox. However, if you made a mistake and *do* want to allow people to change the heading, or add more headings and paragraphs in this region, you need to do two things: First, unlock the editable region you created (see page 687); then, select the text *and* tag (the Tag selector [page 22] is the best way to make sure you've selected a tag), and then turn that into an editable region.

When the New Editable Region dialog box appears, type a name for the region (you can't use the same name twice) and then click OK—avoid hyphens in the name (see the Note on page xx). You return to your template, where the name you gave the region appears in a small blue tab above the editable region (see Figure 19-5).

Tip: If you use tables to lay out your pages (see Chapter 7), you'll often assign one table cell as the main area to hold the primary content of the page. For example, in the pages shown in Figure 19-1, the Frequently Asked Question and its answer appear in a single cell on the page. This cell makes a perfect editable region for a template. In the Tag selector, just click the <td> tag associated with that cell and use any of the techniques discussed here to convert the contents of that cell into an editable region.

If you use CSS, on the other hand, you can create a separate <div> tag (see page 313) for the main content area. In this case, select just the contents of the <div> tag, not the tag itself. Here's one instance where you want to avoid the Tag selector (page 22), which selects the entire <div>, tags and all. If you turn the <div> tag into an editable region, it's possible for someone modifying the page later to delete the tag entirely, which could wreak untold havoc on your CSS-based layout.

Fortunately, Dreamweaver has a handy shortcut for selecting just the contents of a <div> tag. Click anywhere inside the <div> tag, and then press Ctrl+A (⌘-A) or choose Edit → Select All. Next, turn this selection into an editable region, and the <div> tags will remain *outside* of the editable region, so no one can inadvertently delete a <div> tag that helps define the basic structure of a page.



Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam,

Figure 19-5:
This page is based on a template called feature, as you can tell from the little tab in the document window's upper-right corner. You can modify editable regions, which are labeled with small tabs. In this example, one editable region is called feature. An additional editable region (named related) appears within a repeating region (labeled repeatRelated) that lets you duplicate editable regions to form a list of items. Optional regions don't have any clear identifier on the page; you can identify them only in the Template properties window, as described on page 681. The title of any page created from a template is also editable. All other parts of the page are locked (circled); you can make changes only to the original template file.

POWER USERS' CLINIC

Under the Hood of Templates

Dreamweaver saves templates as HTML files in the Templates folder inside your current local site folder (see Chapter 15 for information on local sites). Each template bears the file name extension .dwt to distinguish it from regular Web pages.

The program treats files in the Templates folder differently from normal Web pages, so don't save anything but .dwt files there. In addition, since Dreamweaver expects the Templates folder to be in the local root folder of your site, don't move the Templates folder or change its name in any way (don't even change the capital T in Templates, even if you're a low-key type of person). If you do, your templates won't work.

As with Library items, Dreamweaver uses HTML comment tags to indicate the name of the template. If you inspect the HTML code of a template-based document (see Chapter 10), you'll see that, immediately following the opening <html> tag, Dreamweaver inserts a comment tag with the text "InstanceBegin" followed by the location and name of the template. Additional comment tags indicate areas of the page that you can modify, plus special template features like template parameters used for optional regions. For instance, the title of a page based on a template is always editable; its comment tag might look like this:

```
<!-- InstanceBeginEditable
    name="doctitle" -->
<title>My New Page</title>
<!-- InstanceEndEditable -->
```

The first comment indicates the editable region's beginning and also includes the editable region's name. When editing pages based on the template, you can change only the HTML between these comment tags. Everything else on the page is locked, even when you're working in Code view.

You may find that a single editable region is all you need—for example, a single area of the page (a section of a page enclosed by a <div> tag, for example) containing the text for a product review. However, if you need to edit *multiple* areas of a Web page, just add more editable regions to the template. For instance, when you create a template for an employee page, you can create editable regions for the employee's name, telephone number, and photo. If you change your mind and want to lock a region again, select the editable region and then choose Modify → Templates → Remove Template Markup. Dreamweaver removes the code that makes the region editable. You can do the same thing with other types of template regions, like repeating and optional regions.

Warning: You can rename an editable region by clicking the blue tab on the template page and typing a new name into the Property inspector. However, if you've already built pages based on this template, it's not a good idea. Because template-based pages use the name to identify editable regions, Dreamweaver can lose track of where content should go when you rename a region. See Figure 19-18 for a workaround.

Adding a Repeating Region

Some Web pages contain lists of items. A catalog page may display row after row of product information—picture, name, price, and description. An index of Frequently Asked Questions may provide a list of questions and the dates they were posted.

If you were to make a template for either of these pages, you would add an editable region to the area of the page where these lists appear. Just creating an editable region, however, wouldn't give you any ability to enforce (or easily update) the design of these lists, because *everything* within an editable region can be changed.

Fortunately, Dreamweaver provides a pair of template tools to overcome this problem: *repeating regions* and *repeating tables*. Both let you create areas of a page that include editable (and uneditable) regions that can be repeated any number of times (see Figure 19-6).

FREQUENTLY ASKED QUESTION

Hindered by Highlighting

I'm distracted by the tabs and background colors that Dreamweaver uses to indicate Library items and Templates. How do I get rid of them?

When you use Library items or Templates, you see blue tabs and yellow backgrounds to indicate editable regions and Library items. Although these visual cues don't appear in a Web browser, they can still make your page harder to read while you work in Dreamweaver. Fortunately, you can alter the background color of these items and even turn highlighting off altogether.

Choose Edit → Preferences, or press Ctrl+U (⌘-U). In the Preferences Category list, click Highlighting. To change the background color for editable regions, locked regions, and Library items, use the color box (see page 47) or type in a hexadecimal color value. To remove the highlighting, turn off the Show box next to the appropriate item.

Oftentimes, it's useful to keep highlighting on to help you keep track of Library items and editable regions. If you want to turn off highlighting temporarily, simply choose View → Visual Aids → Invisible Elements, or use the keyboard shortcut Ctrl+Shift+I (⌘-Shift-I) to toggle these visual cues off and on. This technique has the added benefit of hiding table borders, layer borders, and image maps, as well as other invisible elements.

Adding a repeating region is similar to adding an editable region. Select the area of the template page you wish to make repeatable, which usually contains at least one editable region. This could be a single list item (the <li> tag), a table row (<tr> tag), or even an entire <div> tag.

Tip: You can make a repeating region that *doesn't* include an editable region. For example, a template for a movie review Web page could include a repeating region that's simply a graphic of a star. A page author adding a new movie review could repeat the star graphic to match the movie's rating—four stars, for example. (There's just one caveat—see the Tip on page 693.)

Next, tell Dreamweaver that the selected elements are part of a repeating region. You can use any of these techniques:

- On the Common tab of the Insert bar (Figure 19-2), select the Repeating Region option from the Templates menu.
- Choose Insert → Template Objects → Repeating Region.
- Right-click (Control-click) the selection and choose Templates → New Repeating Region from the shortcut menu.

When the New Repeating Region dialog box appears, type a name for the region and then click OK. You return to your template, where the name you gave the region appears in a small blue tab above the editable region (see Figure 19-6). (See page 693 for a discussion of using a repeating region when building a new template-based page.)

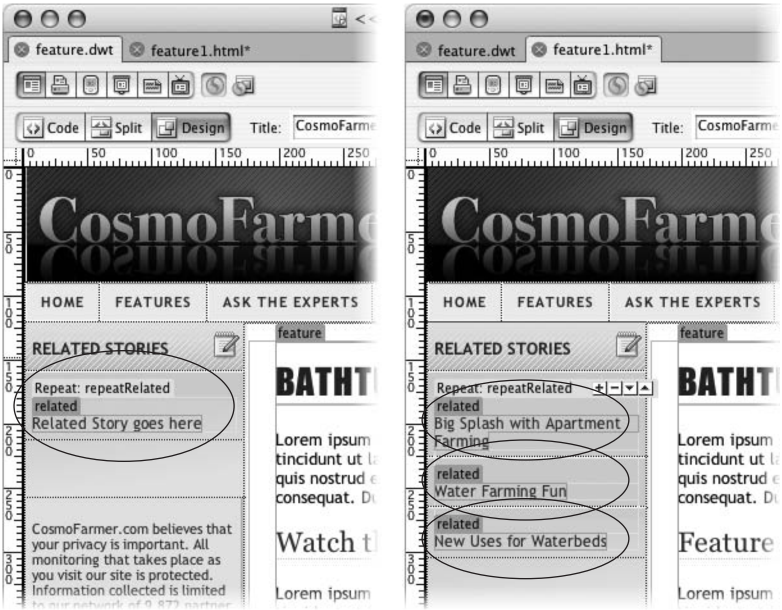


Figure 19-6:
A repeating region lets page authors add multiple selections of repeating information.

Left: In this example, the template has one repeating region, labeled repeatRelated (circled).

Right: A complete page based on this template includes three repeated editable regions (circled). If another page requires more related story listings, you could easily add additional rows to each list. However, the template still controls the basic design. Changing the repeating region's underlying HTML–for example, changing the list items to paragraphs in the uneditable part of the template page–automatically changes the same elements in all pages created from the template. From a design perspective, this strategy also means that page authors can't tamper with the design of a repeating region–just the content marked as editable.

Warning: Dreamweaver lets you name a repeating region with a name already in use by an editable region. But don't–multiple template areas with the same name make Dreamweaver act unpredictably.

Repeating Tables

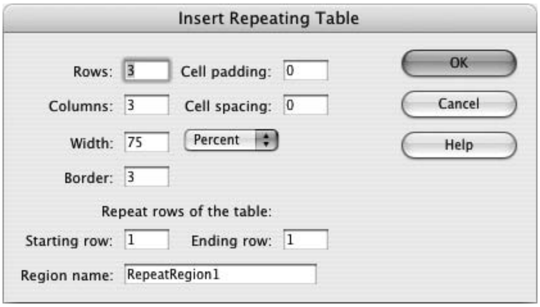
The *repeating table* tool is essentially a shortcut to creating a table with one or more repeating rows. If you had time on your hands, you could achieve the same effect by adding a table to a page, selecting one or more rows, and applying a repeating region to the selection. To use the repeating table tool:

1. **Click the template page where you wish to insert the table.**

   You can't insert a repeating table into an already defined editable, repeating, or optional region, as explained in the box on page 678. You must be in an empty, locked area of the template.

2. **On the Common tab of the Insert bar (Figure 19-2), select the Repeating Table option from the Templates menu.**

   Alternatively, you can choose Insert → Template Objects → Repeating Table. Either way, the Insert Repeating Table window appears (Figure 19-7).



Insert Repeating Table

Rows: 3 Cell padding: 0

Columns: 3 Cell spacing: 0

Width: 75 Percent

Border: 3

Repeat rows of the table:

Starting row: 1 Ending row: 1

Region name: RepeatRegion1

Figure 19-7:
The Insert Repeating Table dialog box lets you kill three birds with one stone: it adds a table to a page, turns one or more rows into a repeating region, and adds editable regions into each table cell inside the repeating region.

3. **Fill out the basic properties of the table.**

   The top part of the window lets you set up the basic structure of the table: rows, columns, cell padding, cell spacing, width, and border. Basically, it's the same information you'd provide when creating any table, as described on page 249. You usually start a repeating table with two rows—one for a heading, and another to contain the information you wish to repeat.

4. **In the "Starting row" box, type the number of the row where the repeating region should begin.**

   Often you'll have just one repeating row: one row of product information, for example. You may want to use the top row for labels indicating the information contained in the rows below. If that's the case, enter *2* at this step, leaving the first row as an uneditable part of the template.

   It's conceivable, however, that you may want each entry to take up *two* rows. The first would list Name and Description; the second would contain a cell for a photo and a cell for the price. You set up this effect in this step and the next.

5. **In the "Ending row" box, type the number of the last repeating row.**

   If you wish to repeat only a single row, enter the same number you provided for step 4. If you're creating a double repeating row, add 1 to the number you provided in step 4. For example, if you need three rows for each repeating entry, add 2 to the number from step 4.

6. **Type a name for this repeating region.**

   Don't use the same name as another template region. You'll run the risk of unpredictable results on template-based pages.

7. **Click OK.**

   Dreamweaver inserts a table into the page. A blue tab with the name of the repeating region appears (see Figure 19-6), as do blue tabs in each cell of each repeated row. These tabs indicate new editable regions—one per cell.

   Since these new editable regions have uninformative names like EditRegion4, you may want to rename them. Click the blue tab and type a new name in the Property inspector. (But do so *before* you create any pages based on the template—see the Warning on page 674.)

To remove a repeating region, select it by clicking the blue Repeat tab, and then choose Modify → Templates → Remove Template Markup. A more accurate way to select a repeating region is to click anywhere inside the region, and then click <mmtemplate: repeat> in the Tag selector (see page 22 for more on the Tag selector). Note that removing a repeating region doesn't remove any editable regions you added inside the repeating region. If you want to rename a repeating region, heed the Warning on page 674.

FREQUENTLY ASKED QUESTION

Editable Regions, Repeating Regions, and Errors

When I try to insert an editable region inside a repeating region, I get the following error: "The selection is already in an editable, repeating, or optional region." *What's that about?*

This error message essentially means you're trying to add a template region where it doesn't belong. It appears most often when you attempt to put a repeating or optional region inside an editable region. That kind of nesting is a no-no; anything inside an editable region can be changed on template-based pages, and as such, Dreamweaver can't touch it.

However, you may get this error message seemingly by mistake. For instance, it's perfectly OK to add an editable region inside a repeating region, and it's even OK to add a repeating region inside an optional region, and vice versa. But one day you select text inside a repeating region and try to turn it into an editable region, and boom–error message. What probably happened was, when you selected the text, Dreamweaver actually selected part of the hidden code used to define a template region (see the box, "Under the Hood of Templates" on page 674) and thought you were trying to put an editable region inside it. To avoid confusion, use the Tag selector to select the tag you wish to turn into an editable region. In the Tag selector, you can click <p> to select the paragraph inside the repeating region. Alternatively, go into Code view (see page 370), and then select whatever part of the code inside the repeating region you wish to make editable.

Making a Tag Attribute Editable

An editable region lets you change areas of HTML—like a paragraph, image, or entire table—on new pages you create from a template. However, when you're creating a template for others to make pages from down the line, you may want to limit these page authors' editing abilities. You may want to allow budding Web designers to change the source of an image used for a banner advertisement without letting them change the width, height, and class applied to the image. Or, you might want to use templates but still let others assign a class or ID to the <body> tag—a move that's normally forbidden on template-based pages. You can use Dreamweaver's Editable Tag Attribute to specify which *tag* properties your successors can change.

Note: Before making a tag attribute editable, first set that property to a default value in the template. Doing so inserts a default value and makes the attribute appear in the Editable Tag Attribute window (see steps 3 and 7 in the following instructions).

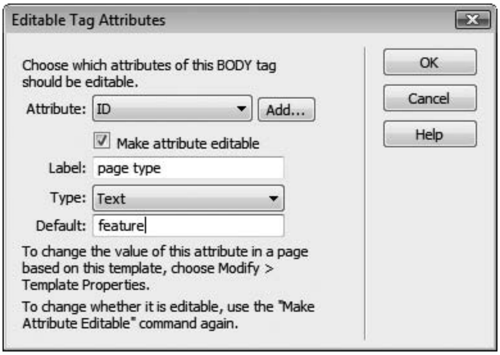
To make a tag attribute editable:

1. **Select the tag whose property you wish to make editable.**

   Using the Tag selector (see page 22) is the most accurate way.

2. **Choose Modify → Templates → Make Attribute Editable.**

   The Editable Tag Attributes window opens (Figure 19-8).



Choose which attributes of this BODY tag should be editable.

Attribute: ID

☑ Make attribute editable

Label: page type

Type: Text

Default: feature

To change the value of this attribute in a page based on this template, choose Modify > Template Properties.

To change whether it is editable, use the "Make Attribute Editable" command again.

Figure 19-8:
Dreamweaver provides detailed control for template pages. To make just a single property of a single tag editable when pages are later based on your template, turn on the "Make attribute editable" checkbox. In this case, the "id" attribute of the body tag is editable, allowing page designers the freedom to apply different CSS styles to the body of each template-based page.

3. **Select an attribute from the menu or add a new attribute with the Add button.**

   Only properties you've already set for the selected tag appear in the Attribute menu. In other words, if you've selected an image, you probably see the *Src*, *Width*, and *Height* properties listed. But unless you've set the image's border, the *Border* property doesn't appear.

   To add a property, click the Add button. In the window that appears, type the appropriate property name. For example, to make the *alt* (alternative text) attribute of a graphic editable, you'd set the <img> tag's *alt* attribute by typing *alt* here. (If you're not sure of the attribute's name, check out Dreamweaver's built-in HTML reference, described on page 394.)

Note: If you want page editors to be able to change a CSS class or ID applied to the <body> tag on template-based pages–to apply different fonts, background colors, or any of the many CSS formatting options to each template-based page–you *have* to make the Class attribute editable. (See page 114 for more on CSS classes.)

4. **Make sure the "Make attribute editable" box is turned on.**

   If you decide that you no longer want to allow editing of this property, you can return to this dialog box and turn off editing, as described on the opposite page.

5. **Type a name in the Label field.**

   What you type here should be a simple description of the editable tag and property, which helps page authors correctly identify editable properties. For example, you could use *Product Image* if you're making a particular image's *source* (Src) property editable.

6. **Choose a value type from the menu.**

   Your choices are:

   - **Text.** Use this option when a property's value is a word. For example, you can change the image tag's *Align* property to *top, middle, baseline*, and so on. Or, when using Cascading Style Sheets, you could make a tag's *Class* property editable to allow page authors to apply a particular custom style to the tag—*content, footer*, and so on.
   - **URL.** Use this option when the editable property is a path to a file, like an image's *Src* property or a link's *Href* property. Using its site management tools, Dreamweaver keeps track of these paths and updates them when you move your pages around your site.
   - **Color.** If the property requires a Web color, like a page's background color, select this option. This option makes Dreamweaver's color box available to people who build pages from the template.
   - **True/False.** You shouldn't use this option. It's intended for Dreamweaver's Optional Regions feature (discussed on the opposite page), and it doesn't apply to *HTML* properties.
   - **Number.** Use this choice for properties that require a numeric value, like an image's *Height* and *Width* properties.

7. **Type a default value into the Default field.**

   This step is optional. The default value defines the initial value for this property, when people first create a page based on the template. They can then modify this value for that particular page. If you've already set this property in the template, its value automatically appears in this box.

8. **Click OK to close the window.**

   Dreamweaver adds code to the template page that allows page authors control of the attribute. Setting this attribute on pages created from the template is described on page 694.

If you later decide that you *don't* want a particular tag's property to be editable, Dreamweaver can help. Open the template file, select the tag with the editable attribute, and choose Modify → Templates → Make Attribute Editable. In the window that appears, turn off the "Make attribute editable" checkbox (Figure 19-8). Unfortunately, doing so doesn't remove *all* of the template code Dreamweaver added. Even after you turn off editing for an attribute, Dreamweaver leaves behind the parameter used to control the tag's property. To eliminate *this* extra code, see the box on page 688.

Adding Optional Regions

Templates provide consistent design. While consistency is generally a good thing, it can also get boring. Furthermore, there may be times when you'd like the flexibility to include information on some template-based pages but not on others.

Dreamweaver provides a fairly foolproof way to vary page design: *optional regions*. An optional region is simply part of a template page that you can hide or display on each template-based page (see Figure 19-9). When creating a new page based on the template, a page author can turn the region on or off.

Creating an optional region is a snap. Just select the HTML code you wish to make optional, and then do one of the following:

- On the Common tab of the Insert bar (Figure 19-2), select the Optional Region option from the Templates menu.
- Choose Insert → Template Objects → Optional Region.
- Right-click (Control-click) the selection and choose Templates → New Optional Region from the shortcut menu.

In the New Optional Region window, type a name (Figure 19-10). Make sure not to use the same name as any other region on the page, and—although Dreamweaver allows it—don't use spaces or other punctuation marks. (Following the rules for naming files as described on page 559 is the best method, and ensures that the optional region works properly.) Click OK to close the window and create the new optional region. Dreamweaver adds a light blue tab with the word "If," followed by the name you gave the region (Figure 19-9).

Locking Optional Regions

An optional region can include editable and repeating regions, *and* locked regions. For example, if you simply want to allow a page author to turn on or off a graphic ("This item on sale!!!!"), insert the graphic outside an editable region on the page,

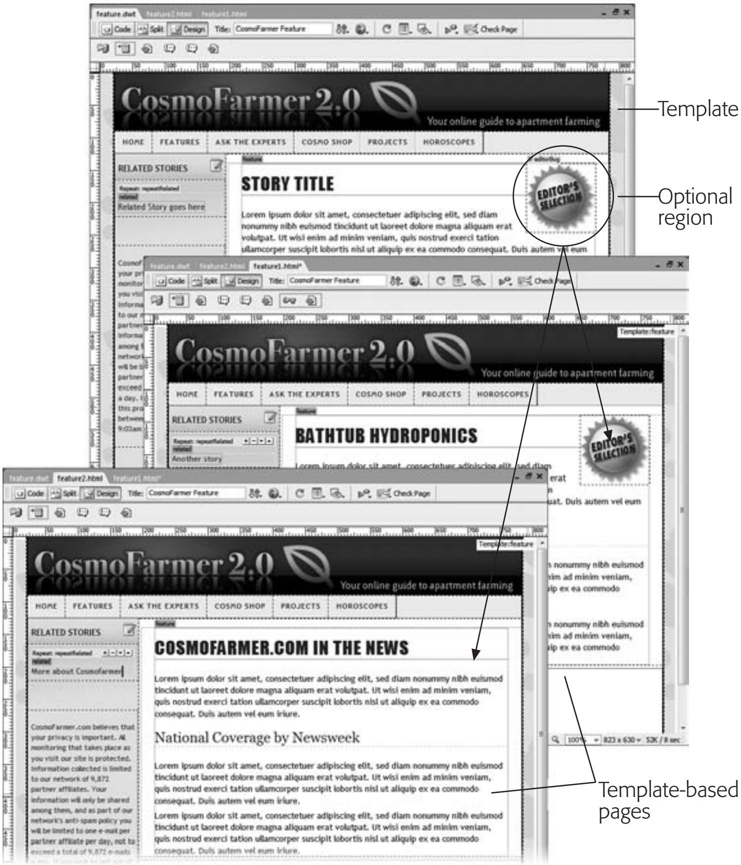
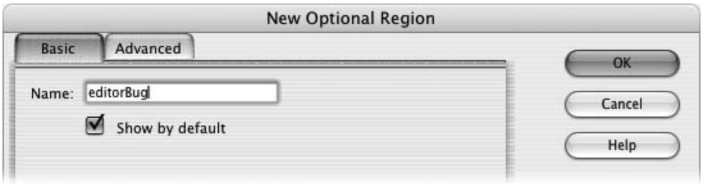


Figure 19-9: *Now you see it, now you don't. Optional regions let you show or hide content on a page-by-page basis. In these examples, the template page (top) has an optional region containing an "Editor's Selection" icon—indicated by the blue tab with the label "If editorBug" (circled). When creating a template-based page from this template, you can either display the optional region (bottom right) or hide it (bottom left).*



Name: editorBug

☑ Show by default

Figure 19-10: *The Optional Regions feature lets you show or hide specific content on template-based pages. Turning on "Show by default" tells Dreamweaver to display the region when a page author first creates a template-based page. Turn this box on if the optional region needs to show on most pages. You'll save someone the effort of turning the region on each time she creates a new template-based page.*

and then make it an optional region as described above. Since anything not inside an editable region is locked, a page author can't change the graphic or ruin its formatting—he can only make it visible or hidden.

Repeating Optional Regions

An optional region can also include repeating regions. For example, suppose you create a repeating region (see page 674) that lets a page author add row after row of links to a list of related articles. You could then turn this repeating region into an optional region, as described above, so that if a particular page had no related articles, the author could simply hide the entire "related articles" section of the page.

Optional Editable Regions

Dreamweaver's Optional Editable Region command inserts an optional region with an editable region *inside* it. To use it, click in the template at the spot where you'd like to add it, and then choose Insert → Template Objects → Optional Editable Region (alternatively, you can choose this option from the Templates menu on the Common tab of the Insert bar). The New Optional Region window appears; give it a name, and then follow the same steps outlined previously for an optional region.

This technique doesn't offer a lot of control; it's hard to insert HTML *outside* the editable region, for example. So if you want to have an image or table that's optional but *not* editable, it's usually better to just create the editable region as described on page 671 and turn it (and any other HTML you wish to include) into an optional region.

Note: The Optional Editable Region command doesn't let you name the editable region; instead you get a generic name like *EditRegion7*. You can select the editable region and change its name in the Property inspector, but do so *before* you build any pages based on this template (see the Warning on page 659).

Advanced Optional Regions

A basic optional region is a rather simple affair: It either appears or it doesn't. But Dreamweaver offers more complex logic for controlling optional regions. For example, maybe you want several different areas of a page to be either hidden or visible at the same time—perhaps an "On Sale Now!" icon at the top of the page *and* a "Call 1-800-SHIZZLE to order" message at the bottom of the page. When one appears, so does the other.

Because these objects are in different areas of the page, you have to create two separate optional regions. Fortunately, using Dreamweaver's advanced settings for optional regions, you can easily have a single region control the display of one or more additional areas of a page. Here's how to do it:

1. **Create the first optional region using the steps on page 681.**

   Give the region a name using the Basic tab of the New Optional Region window (Figure 19-10).

2. **Select the part of the page—an image, paragraph, or table—that you wish to turn into a second optional region.**

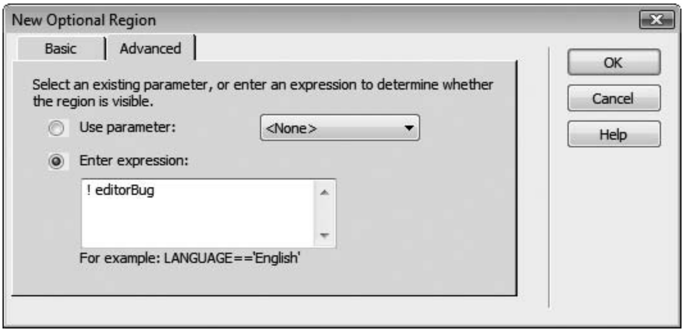
   In this case, you make the display of this region dependent on the optional region added in step 1. If the first region is visible on the page, this second region also shows.

3. **On the Common tab of the Insert bar** (Figure 19-2), **choose the Optional Region item from the Templates menu.**

   The New Optional Region window appears.

4. **Click the Advanced tab.**

   The optional region's advanced options appear (see Figure 19-11). In this case, you want the first optional region you created to control the display of this new region. So instead of giving this region a name, you'll simply select the name of the first optional region in the next step.



Select an existing parameter, or enter an expression to determine whether the region is visible.

○ Use parameter: <None>

● Enter expression:

```
! editorBug
```

For example: LANGUAGE=='English'

Figure 19-11:
The New Optional Region box lets you more precisely control the display of an optional region. You can make the region appear only when another region is visible, or use Dreamweaver's template expression language to create a more complex behavior. In this case, the selected region appears only when another region–named editorBug–is not visible (the ! is a programming equivalent of "is not").

5. **Click the "Use parameter" button and select the name of the first optional region from the menu.**

   This step is what makes the first optional region control this region. If a page author displays the first region, this second region also appears.

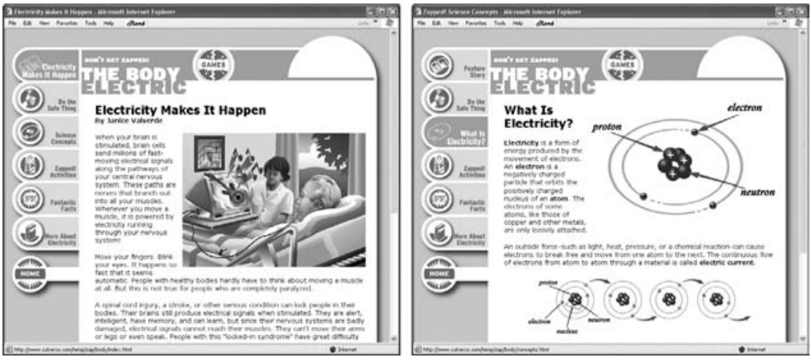
6. **Click OK to close the window and create the new optional region.**

   You can continue adding optional regions in this way, using the Advanced tab and selecting the name of the first optional region from the menu. This way, a single region can control the display of many other areas of the page.

Even fancier tricks

You can use these advanced controls for even more elaborate Web page stunts. For example, say your site is composed of several sections. When a visitor is in one section of the site, its navigation button is attractively highlighted and a secondary navigation bar miraculously appears, offering links to other pages in that section.

Using a template, you can add an optional region containing the highlighted section button. When you add the secondary navigation bar to the page, you make *it* an optional region controlled by the highlighted navigation button. Then, when you add a page to that section of the site, you simply show the optional region containing the highlighted button, causing the secondary navigation bar to appear as well (see Figure 19-12 for a look at how this works).



Electricity Makes It Happen
By Janice Valverde

When your brain is stimulated, brain cells send millions of fast-moving electrical signals along the pathways of your central nervous system. These paths are nerves that branch out into all your muscles. Whenever you move a muscle, it is powered by electricity running through your nervous system!

Move your fingers. Blink your eyes. It happens so fast that it seems automatic. People with healthy bodies hardly have to think about moving a muscle at all. But this is not true for people who are completely paralyzed.

A spinal cord injury, a stroke, or other serious condition can lock people in their bodies. Their brains still produce electrical signals when stimulated. They are alert, intelligent, have memory, and can learn, but since their nervous systems are badly damaged, electrical signals cannot reach their muscles. They can't move their arms or legs or even speak. People with this "locked-in syndrome" have great difficulty

http://www.culverco.com/temp/zap/body/index.html

What Is Electricity?

Electricity is a form of energy produced by the movement of electrons. An **electron** is a negatively charged particle that orbits the positively charged nucleus of an **atom**. The electrons of some atoms, like those of copper and other metals, are only loosely attached.

An outside force–such as light, heat, pressure, or a chemical reaction–can cause electrons to break free and move from one atom to the next. The continuous flow of electrons from atom to atom through a material is called **electric current**.

http://www.culverco.com/temp/zap/body/concepts.html

Figure 19-12:
An optional region on the page at left highlights the top navigation button. By turning on a different optional region (right), the navigation system can highlight the site's current section—"What is Electricity?" (the third button from the top).

Controlling regions with expressions

You can program even more complex behaviors using a basic *expression language*, loosely based on JavaScript, that Dreamweaver understands. For example, instead of having an optional region appear when another optional region is visible (as in the above example), suppose you want to have a region appear when another region is invisible. This arrangement can come in handy when you're creating a navigation bar. When a page is in a particular section, for instance, the navigation button for that section is highlighted, but the button isn't highlighted if the page is in another section.

In other words, you can build a single template for all the sections of a site, but control the appearance of the navigation bar separately for pages in each individual section (see Figure 19-12).

Here's how you'd control the navigation bar:

1. **Click the page where you wish to insert the navigation buttons.**

2. **Insert the highlighted ("You are in this section") navigation button.**

   This button could be a rollover image (see page 229),or just a single graphic. If you have multiple pages in the section, you probably also want to link this graphic to the main page for that section.

3. **Click next to the highlighted button and insert the plain ("You can go here") navigation button.**

   The button could also be a rollover image with a link to the main page for this section (for example, the main Products page).

4. **In the Property inspector, select the highlighted navigation button and link (if it has one).**

   This button appears on any template-based page for this section.

5. **On the Common tab of the Insert bar (Figure 19-2), choose Optional Region from the Template menu.**

   The New Optional Region window appears. Make sure the Basic tab is selected.

6. **Type the name of the section into the Name field. Click OK.**

   For example, if this section of your site advertises your company's products, you can call it *products*. Don't use any spaces or punctuation other than hyphens (-) or underscores (_) for the name. Also make sure the "Show by default" box is *not* turned on. Since you'll be building template-based pages for all the sections of your site, most pages you build will be in other sections of the site. Your work goes faster if this highlighted button starts out hidden. In the next steps, you'll make the plain navigation button appear by default.

7. **Use the Property inspector to select the plain button and link, and then click the Optional Region button on the Insert bar.**

   The New Optional Region window appears again, but this time you'll use the advanced options.

8. **Click the Advanced tab; select "Enter expression" (Figure 19-11).**

   You're going to type an *expression* in the Expression field. An expression is a programming statement that is either true or false. (For an obvious example, 2 is always equal to 2, but it's obviously false to say, "2 is equal to 4." In programming, you express equality using a pair of = signs. So 2==2 is true, but 2==4 is false.) The important thing to remember here is that when an expression is true, the optional region is visible; when it's false, it's hidden.

9. **Type an exclamation point (!) followed by the name you entered in step 6—*!products*, for example.**

   Dreamweaver's template expression language is based on the JavaScript programming language. An exclamation mark means "not," so this code means "*not products*." Translation into non-propeller-head language: when the *products* region (remember, that's the highlighted button) is *not* displayed, this region (button) appears on the page.

   The logic gets a little complicated, but have faith. When you add a new page based on this template, the optional region you added in step 6 is *not* visible (because you turned off the "Show by default" box). In other words, because the region—products in this example—is *not* showing, this region, the one with the plain navigation button, by default appears on the page. Turning the *products* region on (as described on page 694), *hides* the plain navigation button. In other words, the first optional region works like a light switch, alternately turning on one or the other navigation button.

10. **Click OK to close the window and add the additional optional region.**

    Repeat this process for each button in the navigation bar. Now your template is perfectly suited for displaying customized navigation bars for each section of your site. When you create a new template-based page, simply turn on the region for the particular section in which the page is located. (Hiding and showing optional regions is described on page 694.)

As you can see, optional regions are very powerful—and potentially confusing. But using even basic optional regions, you can exert a great deal of control over your template-based pages. For more information on template expressions and optional regions, take a look in Dreamweaver's built-in Help system. (Choose Help → Dreamweaver to open the Adobe Help system; then, in the search box, type *template expressions*, and then hit Enter (Return). The Dreamweaver Help system then lists several articles related to templates and template expressions [as well as a few articles completely unrelated to the topic!].)

Editing and Removing Optional Regions

After inserting an optional region, you can always return to the New Optional Region dialog box to change the region's name, alter its default settings, or use advanced options. To edit an optional region, first select it using one of these techniques:

- Click the region's blue tab in the document window (Figure 19-9).
- Click anywhere inside the optional region in the document window; click the <mmtemplate:if> tag in the Tag selector (see page 22 for details on the Tag selector).

When you select an optional region, an Edit button appears in the Property inspector. Click it to reopen the New Optional Region window. You can then change the region's properties.

To remove an optional region, select it using one of the techniques listed previously and choose Modify → Templates → Remove Template Markup. Dreamweaver removes most of the code associated with the optional region (but see the box "Understanding Template Parameters" on page 688).

Nested Templates

Large sites may have many different sections or types of pages. Each section of the site or type of page may have its own unique look. A Frequently Asked Questions page may have distinct areas for a question, an answer, and links to further resources, while a product page may have a picture, a product description, and ordering information. You could create different templates for each type of page, but even that may be more work than necessary.

While many pages in a site may have subtle differences, they usually share very basic design features. The overall structure of every page, for example, may be the

POWER USERS' CLINIC

Understanding Template Parameters

When you insert an optional region, Dreamweaver adds special code to the head of the Web page. Called a *template parameter*, this code is responsible for showing or hiding an optional region.

In fact, Dreamweaver uses parameters when you make a tag attribute editable, too. A typical parameter for an optional region might look like this:

```
<!-- TemplateParam name="SaleBug"
type="boolean" value="true" -->
```

The <!-- and --> are HTML comments that hide this code from a Web browser. TemplateParam tells Dreamweaver that the comment is actually part of the program's Template features–specifically, a template parameter.

A parameter is composed of three parts: name, type, and value. The name is the name you gave the editable region. The type–Boolean–indicates that the value of this parameter can be only one of two options: true or false. In this example, the value is "true," which simply means that the optional region called SaleBug is visible. (Don't worry; you don't have to actually edit this code by hand to turn optional regions on and off, as you'll see on page 694.)

In programming jargon, a template parameter is known as a *variable*. In simpler terms, it's just a way to store information that can vary. Dreamweaver reacts differently depending on this value: show the region if the parameter's true, or hide it if the parameter's false.

Editable tag attributes also use parameters to store the values you enter for the tag attribute. For example:

```
<!-- TemplateParam name="PageColor"
type="color" value="#FFFFFF" -->
```

On template-based pages, you can change the value of a parameter used for an editable tag attribute to change that tag's property (see page 694).

Unfortunately, when you delete an optional region from a template, or remove the ability to edit a tag attribute, Dreamweaver always leaves these parameter tags hanging around in the head of the template document. Keeping in mind that Dreamweaver adds these parameter tags directly before the closing </head> tag, you can find and remove unused parameter tags in Code view (see Chapter 10).

same: same logo, banner, and navigation bar. Even the basic layout may be the same. And there lies the problem with creating individual templates for each section: if you need to make a very basic sitewide change, like adding a new button to the site's overall navigation system or altering the banner, you need to edit *each* template individually, adding extra time, effort, and risk of making a mistake.

Good news—Dreamweaver offers a tool to solve just this problem: nested templates. A *nested template* is a template you make from another template, which then becomes the *master* template (see Figure 19-13).

Imagine a basic software company Web site with three sections: Support, Our Products, and Downloads. Each section has its own brand of information and specific layout needs. However, all three sections share the same banner and navigation.

To create a template system for this site, you must first create a very basic template that includes elements (including editable regions) shared by all pages—the master template. You can then create a nested template based on the master. On the nested template, you can add further design refinements and additional editable regions for the areas that can be changed on pages created from the nested template.

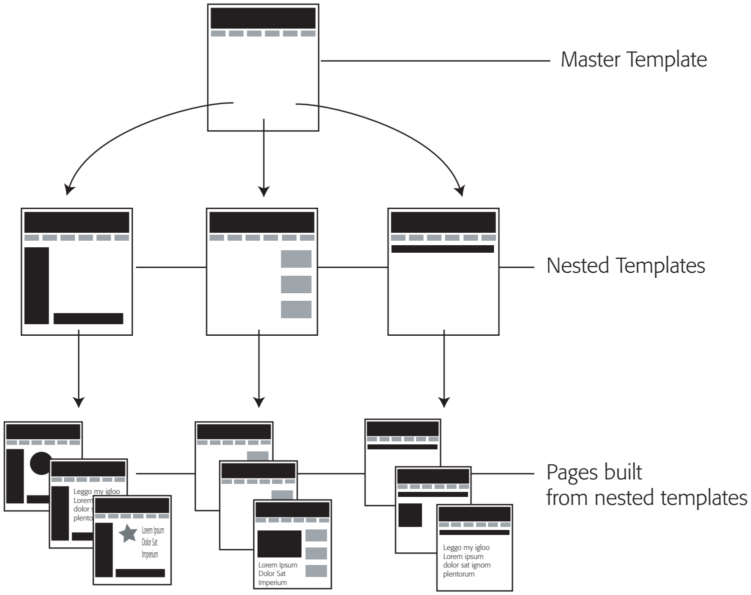


Figure 19-13:
Nested Templates (middle row) let you build templates that share common sitewide design elements while providing precise control for particular types of pages or sections of a Web site. A page built from a nested template (bottom row) contains both elements from a master template (top row)–like a banner and a sitewide navigation bar–in addition to elements specific to its nested template–like a section-specific secondary navigation bar. Changes you make to the master template are passed on to all pages of the site, including nested templates. Changes to a nested template, by contrast, pass on only to pages based on the nested template.

Yes, this process sounds complex—and yes, it is. But when the alternative is hours or days of manual template updating, you can see why serious Web designers are willing to spend the time to master any shortcut they can get.

To create a nested template:

1. **Build a template as described on page 701.**

   This page acts as the master template and controls all nested templates. It should include the basic elements shared by all nested template pages, like your logo and email links. Now is also the time to add editable regions in the areas the nested templates can change, like table cells to hold blocks of text and images.

2. **Name and save this template (File → "Save as Template"), and then close it.**

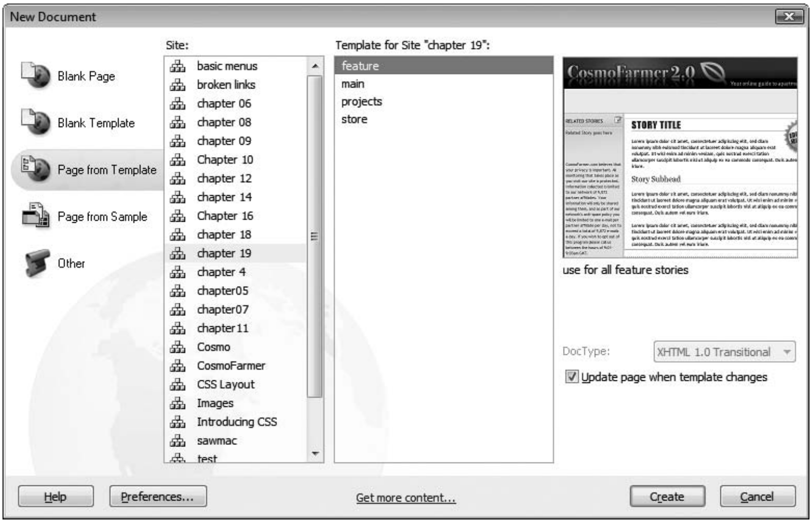
   Your template is safe on the hard drive.

3. **Choose File → New.**

   The window for creating new documents and template-based pages opens (see Figure 19-14).

4. **On the left side of the New Document window, click the "Page from Templates" button. In the Site list, select the Web site on which you're working.**

   You can open templates from any site you've defined in Dreamweaver, but this idea generally isn't a good one, as Figure 19-14 explains.



use for all feature stories

DocType: XHTML 1.0 Transitional

☑ Update page when template changes

***Figure 19-14:** You can use the "Site" list to choose another site you've defined and reveal the list of templates it uses. However, choosing a template stored in a different site isn't a good idea. Dreamweaver doesn't copy any images on the template to the current site and can't translate relative links correctly. The result is broken links aplenty.*

5. **From the list of templates, select the name of the master template file you created in step 1.**

   Make sure the "Update page when template changes" box is turned on. Otherwise, the nested template doesn't update when you edit the master template.

6. **Click OK.**

   Dreamweaver creates a new template-based page. At this point, it's simply a basic Web page based on the original template. Next, you'll turn it into a *nested template.*

7. **Choose File → "Save as Template." Or, on the Common tab of the Insert bar (Figure 19-2), select Make Nested Template from the Templates menu.**

   The Save As Template window appears (see Figure 19-3).

8. **Type a name for the template and click the Save button.**

   Voilà! A nested template.

Customizing Nested Templates

When you first create a nested template, there's no difference between it and the master template. They share the same design, content, and template regions.

The next step is adding the design elements that are specific to pages built from that template. For example, you can add a special type of table for displaying a product photo, description, price, and other information. This table appears only in pages built from this nested template, not from the master template or any other nested template.

There are a few things you should keep in mind when planning your template development strategy:

- When creating pages from templates, you can add content only to an editable region. That's true not only for template-based pages, but for nested templates, too. If the master template has *no* editable regions, you won't be able to change anything on the nested template created from it.
- When working on a nested template, you can insert an editable region only into an editable region supplied by the master template. For example, say you've created a master template to provide a consistent banner and navigation bar to the site, all in a locked region of the master template. Then you add a large empty area at the bottom of the page and turn it into an editable region that you can customize to make specific layouts for each nested template. After creating a nested template from the master template, you can then add new editable regions to this open area. In fact, you can add any template region—repeating, optional, or editable—to this area.
- If, when working on a nested template, you insert a template region (editable, optional, or repeating) into an editable region supplied by the master template, pages based on the nested template can modify *only* those new regions. The rest of the editable region supplied by the master template isn't editable on the pages based on the nested template.

  Using the example in the previous paragraph, let's say you next add a repeating table to your nested template (see page 676 for more detail about repeating tables). When you create a page based on this nested template, you can change *only* the editable areas marked out in the repeating table. Of course, the other side of the coin is that if you add an editable region to the master template and then refrain from adding any particular template regions, all the HTML inside that region is editable in the nested template *and* in all pages based on the nested template.

Using Nested Templates

Here's an example of how you can use nested templates. Suppose you want to create a uniform design for your site where every page of the site has a logo as well as a sitewide navigation bar. Each page within one section of the site also has a sidebar containing a *secondary* navigation bar with navigation buttons for just that section. Finally, every page has a large content area to hold the information specific to that page.

Using nested templates, creating a Web site like this couldn't be easier. Create a master template containing the site banner and navigation bar. This template also includes editable regions for the sidebar and main content area.

Next, create a nested template for one *section* of the site, leaving the content area as it is. Since each page has its own content in this area, you don't need to do anything to this region. Then add the secondary navigation bar to the sidebar area. To lock this region so no one can tinker with the sidebar (in pages built from the nested template), add an empty editable region, or see the Tip on the next page. If you want, you can build similar nested templates for the other sections of the site.

Now you're ready to start building the pages of your site. Create a new page based on one of the section templates. Add text or graphics to the editable content area of the page. Should you need to change the site logo or add a button to the site-wide navigation bar, open the master template, make the changes, save the file, and let Dreamweaver update all the pages of your site with the new look. If you simply need to change the secondary navigation for one section of the site, then open the appropriate nested template, change the sidebar, save the template, and let Dreamweaver update all the section pages.

Tip: You can lock an editable region passed from a master template to a nested template, so that pages based on the nested template can't be changed in this region. In the nested template, go into Code view, and then locate the beginning of the editable region, which looks something like, <!-- InstanceBeginEditable name="regionName" -->. Then insert the text @@("")@@ directly after the -->.

If you find yourself typing this code often, think about creating a snippet (see page 647) containing the text @@("")@@.

Building Pages Based on a Template

Building a template is only a prelude to the actual work of building your site. Once you finish your template, it's time to produce pages.

To create a new document based on a template, choose File → New to open the New Document window (see Figure 19-14). Click the "Page from Template" button, and then, from the Site list, select the current site you're working on. All templates for the selected site appear in the right column. Select the template you wish to use, and then click Create.

Tip: If you don't want your new Web page linked to the template (so that changes to the template also affect the Web page), turn off the "Update page when template changes" checkbox. The result is a new page that looks just like the template, but has no locked regions; you can edit the entire page. This method is useful, for example, when you want to start with the general design and structure of a certain template when creating a brand-new design for another template. (Be aware that Dreamweaver remembers this choice the next time you create a new template-based page. In other words, future pages you create from a template will *also* be unlinked–unless you remember to turn the "Update page" box back on.)

A new Web page document opens, based on the template, bearing a tab in the upper-right corner that identifies the underlying template name. Dreamweaver outlines any editable regions in blue; a small blue tab displays each region's name (Figure 19-5).

Dreamweaver makes it painfully obvious which areas you aren't allowed to edit; your cursor changes to a "forbidden" symbol (a circle with a line through it) when it ventures into a locked area.

To add content to an editable region, click anywhere inside the editable region. You can type inside it, add graphics, or add any other objects or HTML you can normally add to a document. You can also change the document's title and add a Spry Menu bar (Chapter 5), Spry widgets (Chapter 12), Behaviors (Chapter 13), Cascading Style Sheets (see Chapter 4), and meta tag information (items that go in the <head> of an HTML document).

Working with Repeating Regions

Repeating regions work a bit differently than editable regions. In most cases, a repeating region includes one or more editable regions (which you can edit using the instructions above). However, Dreamweaver provides special controls to let you add, remove, and rearrange repeating entries (see Figure 19-15).

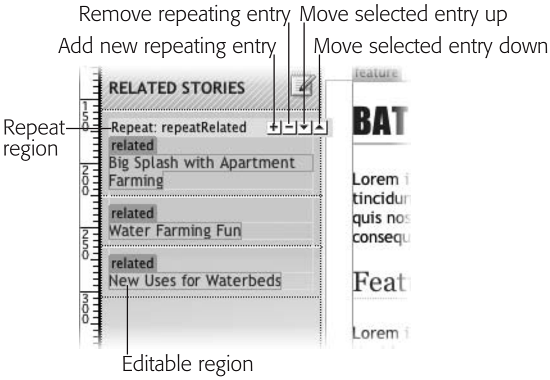


Figure 19-15: *Repeating regions are a great way to quickly add lists to your Web pages. Here, a list of links points to stories related to this Web page. Clicking the + button adds another row to this table, complete with an editable region for adding a new story title and link.*

These regions are intended to let a page author add repeated page elements—like rows of product information in a list of products. To add a repeating entry, click the + button that appears to the right of the Repeat region's blue tab. You can then edit any editable regions within the entry. Click inside an editable region inside a repeating entry and click + again to add a new entry *after* it.

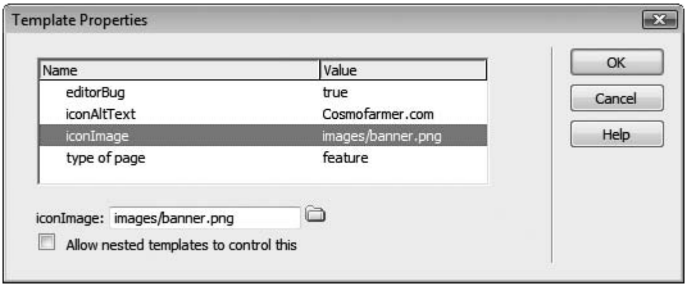
Deleting a repeating entry is just as easy. Click inside an editable region within the entry you wish to delete and click the - sign button.

Tip: You can create repeating regions that don't have any editable regions–for example, repeating a star several times to indicate the rating for a product. Although you can use the + button to repeat such regions, you can't delete those regions with the minus sign (-) button. In other words, you're stuck with any extras you've added. The only workaround is to add an editable region to the repeating region. Then Dreamweaver lets you remove any repeating regions you wish.

To rearrange entries in the list, click inside an entry's editable region. Click the up or down arrows to move the entry up or down in the list (to alphabetize it, for example).

Changing Properties of Editable Tag Attributes

Unlike editable or repeating regions, an editable tag attribute isn't immediately apparent on template-based pages. There's no blue tab to represent it, as there are for editable regions; in fact, nothing appears in Design view to indicate that there are *any* editable *tag* properties on the page. The only way to find out is to choose Modify → Template Properties to open the Template Properties dialog box (see Figure 19-16).



Name	Value
editorBug	true
iconAltText	Cosmofarmer.com
iconImage	images/banner.png
type of page	feature

iconImage: images/banner.png

□ Allow nested templates to control this

Figure 19-16: *The Template Properties window lets you control editable tag attributes and other parameters for optional regions. Depending on which parameter you select, the options at the bottom of the window change. In this case, the SRC property of an image tag has been made editable. You can click Dreamweaver's familiar "Browse for File" button to change the image tag's SRC property by selecting a new graphic file to display on the page.*

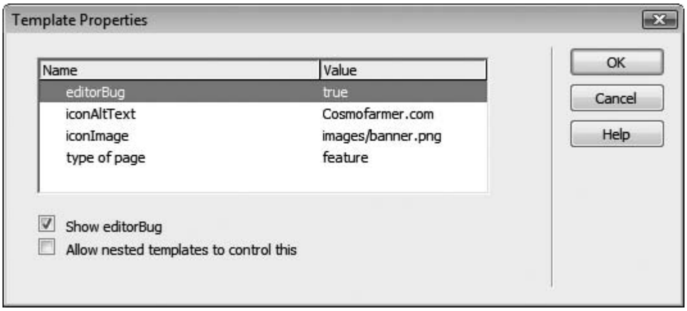
All editable tag attributes for this page appear in this window. In addition, all parameters defined for this page, including optional regions, appear here, as discussed in the box on page 688.

To change the value of a *template* property—in other words, to edit the property of an editable tag—select its name from the list and fill out the option that appears at the bottom of the window. For example, in the case of *color* properties, use the color box to pick a Web-compatible color. If the property is a path (like a link or an image's *source* property indicating the graphic file's location in the site), click the "select a file" folder icon to browse to select the file.

Once you've finished setting the editable properties for the page, click OK to close the window.

Hiding and Showing Optional Regions

As with Editable Tag Attributes, you use the Template Properties window to control the display of optional regions. On template-based pages, you can show or hide an optional region by choosing Modify → Template Properties to open this dialog box (see Figure 19-17). Next, select the name of the optional region. To make all page elements in the region appear, turn on the "Show" checkbox at the bottom of the window. To hide the optional region, turn off this box.



Name	Value
editorBug	true
iconAltText	Cosmofarmer.com
iconImage	images/banner.png
type of page	feature

☑ Show editorBug
□ Allow nested templates to control this

Figure 19-17: *The Template properties window displays optional regions as well as editable tag attributes. A Template property for an optional region–in this example, "editorBug"–has a value of either true or false. True lets people see the contents of the region on the page, while false hides the region. (The "Allow nested templates to control this" option is described in the box below.)*

FREQUENTLY ASKED QUESTION

Controlling the Nest

The Template Properties dialog box includes a checkbox labeled "Allow nested templates to control this." What does it do?

Imagine that you create a template and add several optional regions and editable tag attributes to it. You then use this template as a basic design for more refined templates for each section of your site. When you create one of these nested templates based on the master template, it has access to the Template Properties window, where page authors can modify any of the *template* properties created by the original, master template.

For example, to better identify each section of a site, you might assign a different class style to the <body> tag of a section's pages. The class might apply a different background color to each page within a section: blue for the products section, orange for the support section, and so on. In the master template, you make the <body> tag's *class* property editable. Now, when you create a nested template for the products section, you simply open the Template Properties dialog box, and then assign a class style with a blue page background. For the support section's nested template, apply a class that sets the background to orange. Now, when you create a template-based page for the support section, its background is orange, while a page for the products section has a blue background.

However, to let your site's color palette go really wild, you may want every page in the site to have its own unique background color, each defined by a different class style. (Disclaimer: Don't try this at home.) In this case, you'd want to let every page based on a nested template have an editable *class* property.

To do so, open the nested template, open the Template Properties window, select the property that should be editable in pages built from this template (*color* in this case), and turn on the "Allow nested templates to control this" checkbox. Now this property is uneditable in the nested template, but editable in all pages created from it.

You've probably realized by now that the phrase "Allow nested templates to control this" doesn't make much sense. Turning it on actually prevents the nested template from controlling the property. A better way to think of it is "Allow pages created from this template to control this property."

The bottom line: Turning on this box makes the attribute uneditable on that page. If it's a nested template, it lets the *Template* property "pass through" to all pages based on this template. In other words, you can't set the background color in the template, but page authors can change it in pages created from the template.

Applying Templates to Existing Pages

What happens if you create a Web page, and *then* decide you want it to share the look of a template? No problem. Dreamweaver lets you apply a template to any Web page in your site. You can even swap one template for another by applying a template to a page that's already based on a different template

To apply a template to a page you've already created:

1. **Choose File → Open to open the page you want to alter.**

   The Web page opens.

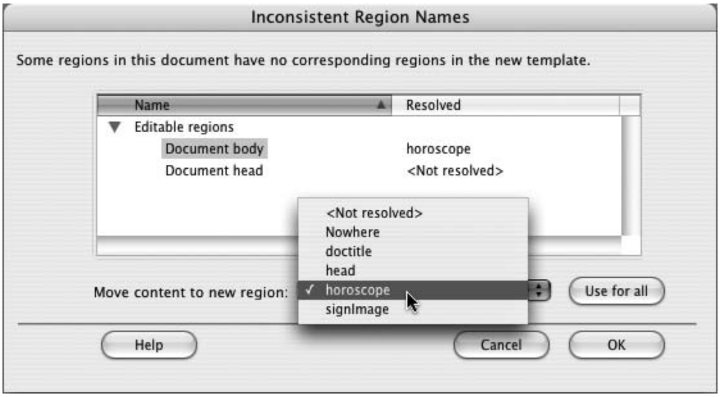
2. **Choose Window → Assets. Click the Assets panel's Templates button (see Figure 19-4).**

   The Assets panel appears and reveals a list of the site's templates.

Tip: You can also apply a template to a page by choosing Modify → Templates → "Apply Template to Page." Select the name of the template from the window that appears and skip to step 5.

3. **Click a template in the list on the Assets panel, and then click Apply.**

   The Inconsistent Region Names dialog box opens (Figure 19-18).



Inconsistent Region Names

Some regions in this document have no corresponding regions in the new template.

Name	Resolved
▼ Editable regions	
Document body	horoscope
Document head	<Not resolved>

Move content to new region: horoscope

Figure 19-18: *When you apply a template to a page you've already created, you must tell Dreamweaver what to do with the material that's already on the page. Tell it what to do by selecting one of the template's editable regions from a pop-up menu, which takes charge of all editable regions in your page.*

4. **In the list under "Editable regions," choose "Document body."**

   To the right, in the Resolved column, you see <Not resolved>. This is Dreamweaver's way of saying it doesn't know what to do with the contents of the current page. You need to pick one of the template's editable regions.

5. **From the "Move content to new region" menu, select an editable region.**

   If you want to keep the material, select the name of an editable region in which to place it from the list; otherwise, choose Nowhere, which, in effect, creates a new blank page based on the template.

Unfortunately, you can only select a single editable region. If the original has several content regions, then Dreamweaver merges them all into a single editable region.

6. **If "Document head" also appears in the window, select it and choose "head" from the "Move content to new region" menu.**

   This step preserves any special information you added to the head of your page, like Cascading Style Sheets, meta tags, custom JavaScript programs, and other information that goes in the <head> of the document. Unfortunately, the title of your original page is always replaced with the default title of the template. You have to reenter the title (see page 38) after you apply the template.

Warning: If you apply a template to a page that has Spry Widgets (Chapters 5, 11, and 12) or Dreamweaver Behaviors (Chapter 13) applied to it, be careful when you select this option. If the same behaviors already exist in the template code, Dreamweaver actually makes a duplicate copy of the JavaScript code in the <head> of the page. To get rid of the extra code, you need to go into Code view (Chapter 10) and manually remove it.

7. **Click OK.**

   Your new page appears.

Updating a Template

Templates aren't just useful for building pages rapidly; they also make quick work of site updates. Pages created from templates maintain a link to the original template file; you can automatically pass changes to a template along to every page built from it. If you used templates to build your site, you probably won't cry on your keyboard when the boss says you must add an additional button and link to the navigation bar. Instead of editing every page, you can simply open the template file, update the navigation bar, and let Dreamweaver apply the update to all the pages.

You update a template (and all the pages based on it) like this:

1. **Choose Window → Assets.**

   The Assets panel appears.

2. **Click the Templates button (see Figure 19-4).**

   A list of the site's templates appears.

3. **Double-click the template's name to open it.**

   Alternatively, you can select the template in the Assets panel, and then click the Edit button to open the original template (.dwt) file (see Figure 19-4).

   The template opens.

Tip: You can also open a template by double-clicking the appropriate template (.dwt) file located in the Templates folder in the Files panel.

4. **Edit the template as you would any Web page.**

   Since this is the original template file, you can edit any of the HTML in the document, including Cascading Style Sheets, meta tags, and layers. You can also add or remove editable regions (see page 674).

   Take care, however, to edit *only* the areas that you did *not* mark as editable regions. The reason: When you update your pages, any region marked as editable in a template file isn't passed on to pages based on that template. After all, the template is supposed to dictate only the design of those pages' *non*-editable regions.

Note: Be careful when you remove editable regions from a template. If you've already built some pages based on the template, Dreamweaver warns you when you save the template. As described below, you can either *delete* the content that was added to that region in each of the pages you created, or move it to another editable region in the page.

5. **Choose File → Save.**

   If you've already created pages based on this template, Dreamweaver opens the Update Template Files dialog box. It lists all the files that use the template.

6. **Click Update to update all files based on the template.**

   Dreamweaver automatically applies the changes you made to the pages based on the template. Then, the Update Pages dialog box opens. If you want to see a list of all files Dreamweaver changed while updating your site, turn on the "Show log" box.

   On a large site, this automatic update feature can be an incredible time-saver, but you may *not* want to click Update, at least not right now. Perhaps you're just saving some of your hard work on the template but aren't quite finished perfecting it—why waste your time updating all those pages more than once? In such a scenario, click the Don't Update button. Remember, you can always update the pages later (see the box on the opposite page).

7. **Click Close.**

   The Update Pages dialog box closes.

Remember that you need to update all your files even if you make a simple change to the template, like changing its name.

Updating Nested Templates

When you build a Web site using nested templates, you have multiple templates affecting your pages. The master template controls design elements of a nested template, which in turn controls pages based on the nested template. (You can even make nested templates *out of* nested templates, but for sanity's sake, you'd better not.) With this level of complexity, updates to nested templates can get confusing fast.

In a nutshell, here's how it works:

- If you edit a locked region in a master template and then update your site, not only does a nested template update, but so do all pages built from it.
- If you edit a locked region in a nested template and then update, those changes pass on to pages built from that nested template.

However, changes you make to an *editable* region of a master template don't pass on to any page. Neither do changes you make in editable regions of a nested template.

Note: Sometimes after making changes to a master template, Dreamweaver doesn't update pages based on nested templates. To safely verify that all template updates are correct, recreate the Site Cache (Site → Advanced → Recreate Site Cache), choose Modify → Templates → Update Pages, and then select the "Entire Site" option.

POWER USERS' CLINIC

Wait to Update

Whenever you modify and save a Library item or a template, Dreamweaver gives you the option of updating any pages in the site that are descended from it. Very often, you'll say Yes.

But there are times when you want to wait to update the site. If you're making a lot of changes to multiple Library items or templates, you may wish to wait until you've finished all your edits before letting the changes ripple through your pages. After all, it can take some time to update large sites with lots of pages.

Dreamweaver lets you update pages that use Library items and templates at any time. Just choose Modify → Library → Update Pages, or Modify → Templates → Update Pages. Both menu options open the same window, the Update Pages dialog box.

At this point, you can update pages that use a specific Library item or template by going to the "Look in" menu, choosing "Files that Use," and then selecting the appropriate name from the pop-up menu. If you would like to update all pages in the site, choose Entire Site, and then, from the pop-up menu, select the name of the local site. Turn on both the "Library items" and Templates checkboxes to update all pages.

To see the results of Dreamweaver's work, check the "Show log" box. This box presents a list of all the updated files.

Unlinking a Page from a Template

If you're confident that you won't be making any further changes to a page's template, and you'd like to be able to edit the page's locked regions, you can break the link between a page and its template by choosing Modify → Templates → "Detach from Template."

All the HTML in the page is now editable, just as on a regular Web page—which is what it is now. You've removed all references to the original template, so changes to the template no longer have any effect on this page.

Note: If you unlink a nested template from its master template, Dreamweaver removes only the code provided by the original master template. Any editable regions you added to the nested template remain.

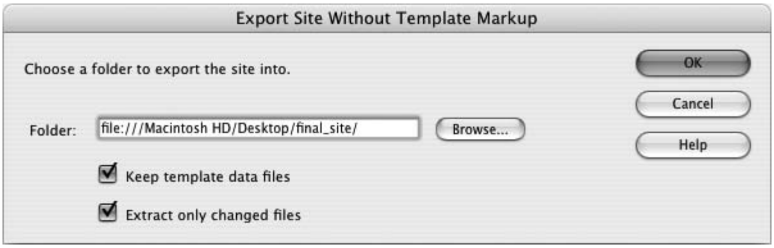
Exporting a Template-Based Site

The good news about Dreamweaver's sophisticated templating features is that they let you build complex Web pages that are easy to create and update. The bad news is that some behind-the-scenes code is necessary to achieve this ease of use. Dreamweaver's template features rely on HTML comment tags to identify editable, optional, and repeating regions, as well as nested template and editable tag attributes (see the box on page 674). Although this code is only for Dreamweaver's use and has no effect on how a Web browser displays the page, it does add a small amount to the size of your Web pages.

Fortunately, Dreamweaver includes a feature that lets you export an entire site into a new folder on your computer *without* any template markup code. It's a good last step before transferring a freshly completed Web site to a Web server.

1. **Choose Modify → Templates → Export Without Markup.**

   Dreamweaver uses the currently active site, so make sure you've got the site you wish to export selected in the Files panel. The Export Site Without Template Markup window appears (see Figure 19-19).



Export Site Without Template Markup

Choose a folder to export the site into.

Folder: file:///Macintosh HD/Desktop/final_site/

☑ Keep template data files

☑ Extract only changed files

Figure 19-19: *Dreamweaver lets you strip out template code from template-based pages with the Export Site Without Template Markup command.*

2. **Click the Browse button, and then select a folder for the exported site.**

   Select a folder *other* than the current local site folder. You always want to keep the original files in the local folder, since they're the ones that keep the template markup, making future updates possible.

3. **Turn on the export options you want.**

   The Export window includes two options. The first, "Keep template data files," creates an XML file for each template-based page. In other words, when you export the site, there's one HTML page (without any special template code) and an XML file (which includes all the template code as well as the page content).

   Theoretically, you could then go back and choose the File → Import → "XML into Template" to recreate the page, complete with the original template information. However, in practice, you probably won't. For one thing, this process creates lots of additional files that you wouldn't want to move to the Web site. Also, when you want to work on the site to update and edit it, you should use the original files in the site's local folder anyway, since they still have the useful template code in them.

The "Extract only changed files" option speeds up the process of exporting a large template-based site. This option forces Dreamweaver to export only pages that you've changed since the last export. Unfortunately, it doesn't tell you *which* files it exported until after the fact. So, to make sure you get those newly exported files to the Web server, you need to keep track of changes by hand.

4. **Click OK to export the site.**

   Dreamweaver goes through each page of the site, stripping out template code and exporting it to the folder you specified.

It's a fine idea to perform an export after you've completed your Web site and are ready to move it to the Internet. You can then move the lean, clean exported files to the Web server.

You can use Dreamweaver's FTP feature to do the uploading (see page 620), but you need to create a new site and define the folder with the *exported* files as a local root folder. Whenever you need to add or update template-based pages, use the original site files, and then export the changed files. You can then switch to the site containing the exported files and transfer the new or updated files to the Web server.

Template Tutorial

In this tutorial, you'll create a template for the CosmoFarmer Web site. Then you'll build a page based on that template and enjoy an easy sitewide update courtesy of Dreamweaver's templates feature.

Note: You'll need to download the tutorial files from *www.sawmac.com/dwcs3/* to complete this tutorial. See the note on page 39 for more details.

Once you've downloaded the tutorial files and opened Dreamweaver, define a new site as described on page 28: Name the site *Templates,* and then select the Chapter19 folder (inside the MM_DWCS3 folder). (In a nutshell: Choose Site → New Site. In the Site Definition window, click the Advanced tab, type *Templates* into the Site Name field, click the folder icon next to the Local Root Folder field, navigate to and select the Chapter19 folder, and then click Choose or Select. Finally, click OK.)

Creating a Template

1. **Open the Files panel by pressing the F8 key.**

   Of course, if it was already open, you just closed it. Press F8 again.

2. **In the Files panel, find and double-click the page *main.html.***

   It's usually easier to start with existing Web page, and then save it as a template. For the purpose of getting to bed before midnight tonight, pretend that you've just designed this beautiful Web page.

3. **Choose File → Save As Template.**

   The Save As Template dialog box opens.

4. **In the description field, type *Use for main site pages*.**

   This description appears in the New Template window when you create a page based on this template.

5. **Name the template *Main*; click Save. In the Update Links window, click Yes.**

   Behind the scenes, Dreamweaver creates a new folder—Templates—in the site's root folder, and saves the file as Main.dwt inside it. A new template is born. You can see it in the Templates page of the Assets panel, as well as the new Template folder in the Files panel.

   The template is a model for other pages. But although they'll be *based* on its design, they won't be identical. The next step is to identify those areas of the design that'll change from page to page—the editable regions.

6. **Drag the "S" in the heading "Story Title" down and to the right until you've selected the two paragraphs and red subhead. Stop just after selecting the final period in the last paragraph.**

   You've just selected all the text in the page's main area. If the "Editor's Selection" graphic in the top right is also highlighted, you've selected too much. That graphic shouldn't be part of the editable region (it's actually an absolutely positioned div, whose HTML code is located *after* all the text in this div).

7. **Choose Insert → Template Objects → Editable Region.**

   The New Editable Region dialog box appears. Here, as in the following steps, you can also, from the Insert bar's Common tab, go to the Templates menu and choose the Editable Region option (Figure 19-2), or just press Ctrl+Alt+V (⌘-Option-V).

8. **Type *Story*; click OK.**

   A small blue tab, labeled *Story*, appears above the headline. So far, you've added one editable region—the most basic type of template region. Next, you'll explore some of the advanced templating features Dreamweaver offers.

9. **On the Web page's top right, click the Editor's Selection button.**

   This step places the cursor inside a <div> tag. You'll turn this div into an optional region so that it can be hidden on most pages, but displayed on special, select pages.

10. **On the top-left corner of the image, click the selection handle to select the div.**

    You can also use the Tag selector at the bottom of the document window: Click <div#selection>. Either way, the div is selected and you're ready to turn it into an optional region.

11. **Choose Insert → Template Objects → Optional Region.**

    The New Optional Region window appears.

12. **Type *editorBug* in the name field, turn off the "Show by Default" checkbox, and then click OK.**

    Since most pages *won't* display this graphic, you'll speed up your work by hiding it by default, and making it visible just on important pages. You see a blue tab labeled "If editorBug" below the text in the main part of the page: this part is the optional region, and represents where the HTML code with the graphic is located—because it's an absolutely positioned div (see page 343), its visual location on the page is different from its code.

    Next, you'll add a repeating region to the left sidebar, so that you can add links to multiple related stories. You'll then make the text editable, so that you can add story titles and links.

13. **In the left sidebar, click the phrase "Related story here."**

    This phrase is enclosed inside a <li> tag. Each time you want to add a new related story, you need to duplicate this tag. To do so, turn it into a repeating region, as follows.

14. **At the bottom of the window, in the Tag selector, click the <li> tag.**

    You've just selected the <li> tag and the text and link inside it. Because any particular story may have many related stories, you'll want to list multiple story titles per page (maybe more on some pages and less on others). So this spot is a perfect place for a repeating region.

15. **Choose Insert → Template Objects → Repeating Region. In the window that appears, type *repeatRelated*. Click OK.**

    Dreamweaver inserts a new repeating region with the familiar blue tab. The tab reads "Repeat: repeatRelated," indicating that it isn't any ordinary template region—it's a repeating region. However, turning a part of the page into a repeating region doesn't automatically make it editable. Since you want to edit the text and add new names to each page, you need to add an editable region *inside* this repeating region.

16. **Triple-click inside the text "Related story here" to select the dummy text and the link. Choose Insert → Template Objects → Editable Region. In the Name field, type *relatedStory*, and then click OK.**

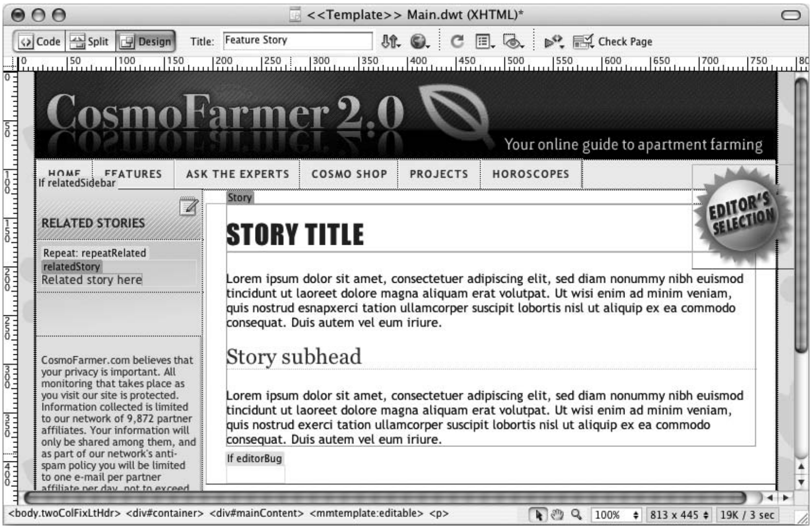
    Another blue tab, labeled "relatedStory," appears inside the repeating region. On template-based pages, you can now change this text, plus add additional story titles, easily. Of course, it's possible that one page may have no related stories; a page would look pretty weird if it had a "Related Stories" headline followed by an empty space. Fortunately, you can turn that headline and the repeating region you just added into *another* optional region. That way, you can show this area on any page that has related stories, but hide it on pages that have no related stories.

17. **In the left sidebar, click inside the headline "Related Stories." In the Tag selector at the bottom of the document window, click <div.related>.**

    This div contains both the headline and the unordered list of story links.

18. **Choose Insert → Template Objects → Optional Region; in the New Optional Region Window's Name field, type *relatedSidebar*, and then click OK.**

    The template should look like Figure 19-20. One last thing: You'll add an editable attribute to the <body> tag, so that you can add a unique ID to each template-based page. The ID defines which section of the site the page belongs to (Features, Projects, and so on).



CosmoFarmer.com believes that your privacy is important. All monitoring that takes place as you visit our site is protected. Information collected is limited to our network of 9,872 partner affiliates. Your information will only be shared among them, and as part of our network's anti-spam policy you will be limited to one e-mail per partner affiliate per day, not to exceed

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud esnapxerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure.

<body.twoColFixLtHdr> <div#container> <div#mainContent> <mmtemplate:editable> <p>

Figure 19-20: *A template file. Depending on your Dreamweaver preference settings, you might also see a couple of gold shield icons representing the absolutely positioned div tag, and the mainContent div that's positioned using relative positioning (see page 338). For instructions on turning those gold shield icons off and on, turn to page 383.*

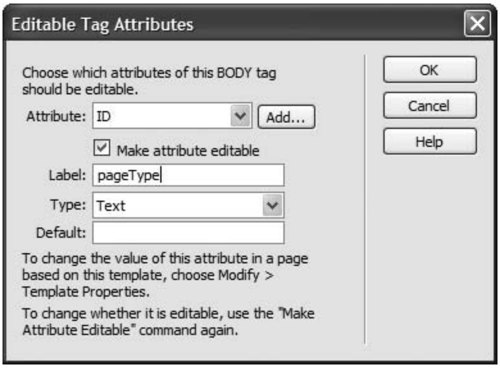
Note: If you're a conscientious computer veteran, you've probably already saved this template by now and noticed an annoying pop-up warning you that the *relatedStory* region is "inside a block tag." This is the same warning discussed on page 678 and below in step 22 on the opposite page. Don't worry about it–it's OK. If it bugs you, just turn on the "Don't show me this message again" checkbox.

19. **In the Tag selector, click <body.twoColFixLtHdr>. Choose Modify → Templates → Make Attribute Editable.**

    The Editable Tag Attributes window appears (Figure 19-21).

20. **Click the Add button, and then, in the pop-up window that appears, type *ID*. Click OK to return to the Editable Tag Attributes window.**

    You've just told Dreamweaver that you'd like to make the <body> tag's ID attribute editable. ID should now appear in the Attribute window.



Editable Tag Attributes

Choose which attributes of this BODY tag should be editable.

Attribute: ID

☑ Make attribute editable

Label: pageType

Type: Text

Default:

To change the value of this attribute in a page based on this template, choose Modify > Template Properties.

To change whether it is editable, use the "Make Attribute Editable" command again.

Figure 19-21:
The Editable Tag Attributes window lets you make a specific property of a selected tag editable on template-based pages—in this case, the ID attribute for the <body> tag. Anyone building pages based on this template can add a unique ID to different pages on the site.

21. **Turn on the "Make attribute editable" checkbox. In the Label field, type *pageType*. Choose Text from the Type menu, and leave the Default value blank.**

    The window should look like Figure 19-20. The label helps you identify which property of which page element you're editing.

22. **Click OK to close the window. Choose File → Save.**

    If you haven't already encountered it (see the Note on the opposite page), Dreamweaver pops up an annoying message informing you that the editable region *relatedStory* is inside a block element (the <li> tag), and that anyone who uses this template can't add some types of content inside this region. Technically, this warning is false. You can add all sorts of content, even tables, divs, and multiple paragraphs inside an <li> tag. This irritating error is Dreamweaver's way of "helping" by pointing out what it believes must be an error on your part. But it's not an error, since you can add additional block-level elements such as headlines to a list item. To make it go away, turn on the "Don't show me this message again, you annoying program" box.

23. **Click OK to close the dialog box. Close this file.**

    Congratulations! You've created your first template.

Creating a Page Based on a Template

Now it's time to get down to business and build some Web pages. Look at the Files panel and make sure you've selected the site that you defined in step 1 at the beginning of this tutorial. Then proceed as follows:

1. **Choose File → New.**

   The New Document window opens.

2. **On the window's far left side, click the "Page from Template" button.**

   A list of all defined sites appears in the Site list.

3. **Make sure the site you defined for this tutorial is selected; also make sure the "Update page when template changes" checkbox is turned on.**

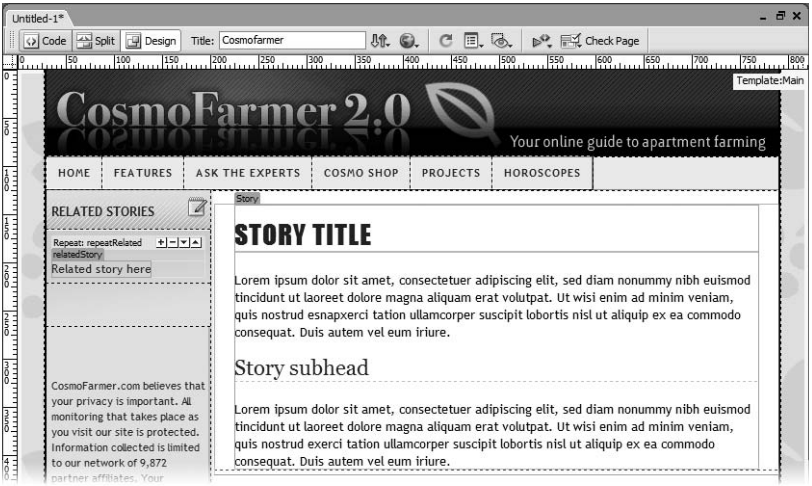
   If you don't turn on the "Update page" box, the new page doesn't link to the original template file—and doesn't update when you make changes to the template.

4. **From the templates list, select Main, and then click Create.**

   And lo, a new, untitled Web page document appears, one that looks (almost) exactly like the template (Figure 19-22).

5. **Choose File → Save. Save the file as *hydroponics.html* in the root folder. At the top of the document window, type *Bathtub Hydroponics* in the title field.**

   To indicate that it's your template's offspring, the document window has a yellow tab in the upper-right corner that reads Template: Main. You can see your editable and repeating regions indicated by blue tabs. Now it's time to add some content.



Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud esnapxerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure.

CosmoFarmer.com believes that your privacy is important. All monitoring that takes place as you visit our site is protected. Information collected is limited to our network of 9,872 partner affiliates. Your

***Figure 19-22:** In template-based pages, blue tabs identify editable areas of the page, and the yellow tab at the top right lists the template's name. Notice that the repeating region has small control buttons (+, -, and up and down arrows) and that one of the optional regions—the "Editor's Selection" graphic—is invisible. (Remember, you deselected the "Show by Default" option for this region.)*

6. **Choose File → Open; in the Open file window, click the Site Root button; navigate to the contents folder, and then double-click the file *feature.html*.**

   You can also open this file by double-clicking its name in the Files panel. The *feature.html* page contains the content for the new template-based page. It's just a matter of copying and pasting the text from one page to the other.

7. **Choose Edit → Select All; choose Edit → Copy. At the top of the document window, click the hydroponics.html tab to switch to the template-based page.**

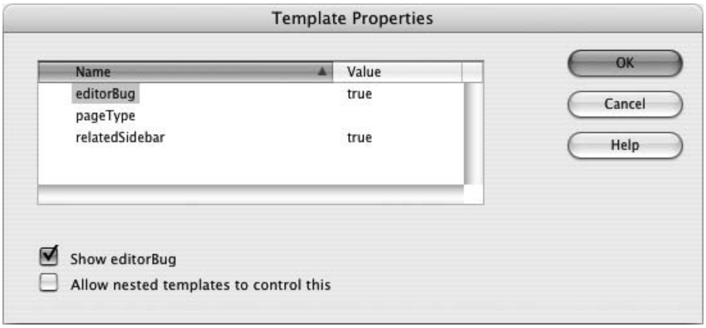
   Remember that you can add content only to an editable region. If you move your mouse over the banner, navigation, and footer areas of the page, you see a black "forbidden" symbol. You can't insert the cursor anywhere but inside an editable region.

8. **Click the Story editable region (in the headline "Story Title," for example). Choose Edit → Select All; choose Edit → Paste.**

   The dummy content is replaced with a CosmoFarmer feature story. It's such an important story that it deserves the "Editor's Selection" seal of approval. That region is one of the optional regions you added to the template. It's not showing up on this page, because in step 12 on page 703, you turned off the "Show by Default" option. To make it appear, you have to turn it on manually.

9. **Choose Modify → Template Properties.**

   The Template Properties window appears (see Figure 19-23). There are three items listed: The first is the optional region you wish to make visible; the second is the editable tag attribute for the <body> tag, and the third is the optional region for the sidebar's related stories area.



Name	Value
editorBug	true
pageType	
relatedSidebar	true

☑ Show editorBug

□ Allow nested templates to control this

Figure 19-23:
The Template Properties window does double duty. Not only does it let you hide or show optional regions, it's also the place to set values for editable tag attributes.

10. **In the properties list, select "editorBug," and then turn on the "Show editorBug" checkbox. Click OK to close the window.**

    The "Editor's Selection" button appears. Next you'll add a list of related stories.

11. **In the left sidebar, click the blue tab labeled "relatedStory."**

    You've just selected the text and the link in that editable region.

12. **Type *Big Splash with Apartment Farming,* and then link that text to the file *splash.html* in the root folder.**

    If you need a refresher on linking, turn to page 159. There are a few other related stories. Fortunately, you can add more entries using the repeating region controls.

13. **Click the + button just to the right of the blue "Repeat: repeatRelated" tab.**

    You've added another row to the page, as shown in Figure 19-24.

Tip: If a page has a lot of elements crowded together–tables, images, text, and so on–Dreamweaver sometimes can't display the small buttons that let you add and remove repeating entries. In this case, you can also use the Modify menu. Click a repeating region, and then choose Modify → Templates → New Entry After Selection to add a new row after the current row, or New Entry Before Selection to add a new row before the current one.

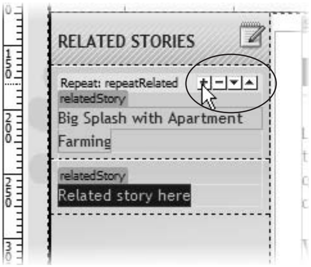


Figure 19-24:
The control buttons to the right of repeating regions (circled) let you add, remove, and rearrange repeating entries. Here, clicking the + button adds another table row for inputting an additional celebrity name.

14. **Click the blue tab of the newly added "relatedStory" editable region, and then type *Water Farming Fun*. Link this text to *water_fun.html* in the site's root folder.**

    For now, you'll add just one more story to the list, though you can actually add as many stories as you want.

15. **Repeat steps 13 and 14 to add one more story: *New Uses for Waterbeds*. Link that text to *waterbeds.html*.**

    The "New Uses for Waterbeds" story happens to be one of the most visited Web pages on the site, so it had better get top billing.

16. **Select "New Uses for Waterbeds" (either select the text, or click the blue tab above the name), and then click the up arrow button next to the repeating region twice.**

    The up and down arrows (Figure 19-24) let you move a repeated region above or below other repeated regions.

    One last thing. Each page within a section of the site needs to be identified with an ID applied to the page's <body> tag. This ID controls particular CSS styles that affect how the navigation bar displays.

17. **Choose Modify → Template Properties.**

    The Template Properties window opens again (see Figure 19-23). You just need to specify an ID name.

18. **From the list of template properties, select pageType. In the pageType text box near the bottom of the window, type *feature*; click the OK button.**

    Notice that the Features button in the navigation bar turns black. This is kind of a "you are here" indicator, identifying the section of the site this page belongs to. (How's this magic done? Descendent selectors, of course. Basically, the link has a class named *.featurelink* applied to it; in the site's style sheet, there's a descendent selector named *#feature a.featurelink*. By setting the ID of the <body> to *feature*, the Features link becomes a descendent of the *#feature id*, and the style *#feature a.featurelink* kicks into action. For more on descendent selectors see page 285.)

19. **Preview this page in a Web browser (F12 [Option-F12]). Return to Dreamweaver, and save and close this Web page.**

    You'll create one more template-based page.

Creating Another Template-Based Page

Templates are useful only if you use them to build lots of pages. You'll build one more template-based page for this tutorial, and see how optional template regions let you create very adaptive Web pages.

1. **Choose File → New. In the New Document window, click the "Page from Template" button; from the templates list, select Main, and then click Create.**

   Another new Web page.

2. **Choose File → Save. Save the file as *planter.html* in the root folder. In the title field at the top of the document window, type *Make an Asparagus Planter*.**

   You'll add some already created content to this page.

3. **Repeat steps 6–8 on page 706. Use the file *project.html* located in the contents folder.**

   This article isn't an "Editor's Selection" (what a surprise) so it's OK to leave that button off this page. However, there aren't any related stories either. Instead of leaving the "Related Stories" section of the left sidebar empty, you'll just turn it off (remember you turned this area into an optional region earlier).

4. **Choose Modify → Template Properties; in the Template Properties window, select the relatedSidebar property; uncheck the "Show relatedSidebar" box.**

   While you're here, you might as well apply the appropriate ID for this page as well.

5. **From the list of template properties, select pageType. In the pageType field, type *project,* and then click OK to close the window.**

   Poof! The Related Stories headline and list disappear. In addition, the Projects button in the navigation bar changes color.

Note: If, at a later time, CosmoFarmer does add a story related to this Web page, the Related stories area can be made visible again, simply by choosing Modify → Template Properties, and then turning on the "Show relatedSidebar" checkbox.

6. **Preview the page in a Web browser.**

   While most of the page looks the same as the first template-based page you created, the use of optional regions let you hide areas of the page without having to create another template. Next you'll edit the template file and update the Web site.

Updating a Template

Now the fun begins. Remember, this page maintains a reference to the original template. In the final phase, you're going to make a few changes to the template. Choose Window → Assets to open the Assets panel, and then click the Template button to reveal the templates for this site (see Figure 19-4):

1. **Return to Dreamweaver, and in the Assets panel, double-click the Main template to open it.**

   The original template, the Main.dwt file, opens. To help market CosmoFarmer's sister publication, The National Exasperator, you'll insert a small ad in the left sidebar.

2. **In the left sidebar, click the empty space just above the paragraph that begins with "CosmoFarmer.com believes that."**

   You'll add an image here.

3. **Choose Insert → Image; select and insert the image *national_ex.gif*, which is located in the images folder.**

   When the Accessibility Options window appears, in the Alternate text field, type *Subscribe to the National Exasperator*. Now you'll just link this image to the National Exasperator Web site. (If you need a reminder on how to insert images, visit page 199.)

4. **In the Property inspector's link box, type *http://www.nationalexasperator.com/*.**

   You could of course continue to edit the template; add another button to the navigation bar, for instance, or update the copyright notice. But making just this one addition is enough to demonstrate the power of Dreamweaver templates, as you'll witness next.

5. **Choose File → Save.**

   The Update Template Files window appears. This is the moment of truth.

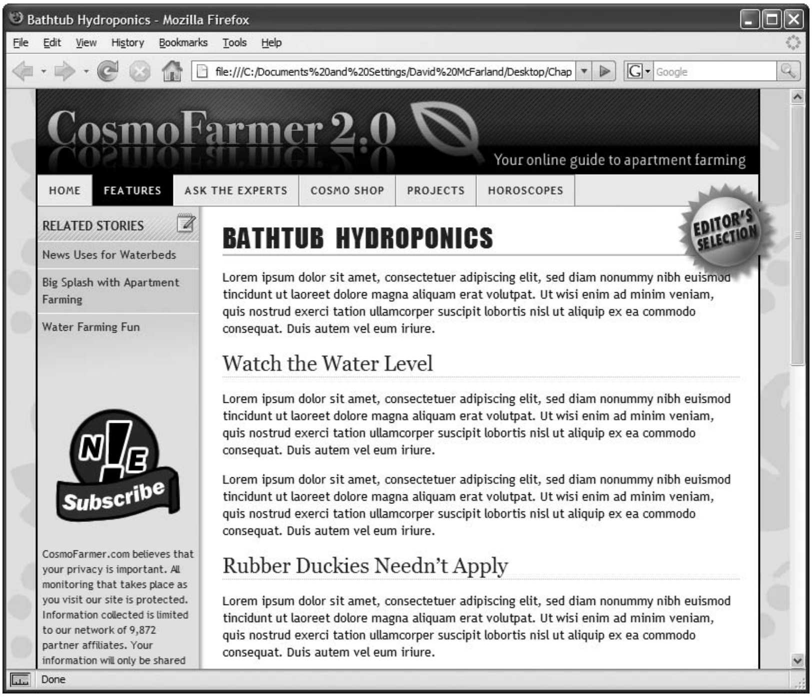
6. **Click Update.**

   Dreamweaver opens the Update Pages dialog box and updates the appropriate Web pages, adding the new image and link to each one. In this case, you based only two pages on the template, so Dreamweaver updates only two pages—as indicated by the list of changes Dreamweaver shows when it's finished.

Note: If, after you update pages based on a template, you don't see the number of updated pages listed in the Update Pages window, turn on the Show Log checkbox.

7. **Click Close to close the Update Pages dialog box. Finally, open the files *hydroponics.html* and *planter.html*.**

   Notice that the National Exasperator graphic has been added to both pages (see Figure 19-25). This series of events happened because you changed the template to which the page was genetically linked. Ah, the power!



BATHTUB HYDROPONICS

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure.

Watch the Water Level

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure.

Rubber Duckies Needn't Apply

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure.

CosmoFarmer.com believes that your privacy is important. All monitoring that takes place as you visit our site is protected. Information collected is limited to our network of 9,872 partner affiliates. Your information will only be shared

Figure 19-25:
The finished tutorial page, complete with a list of related stories, a National Exasperator ad, body copy, and the honorary "Editor's Selection" badge.

CHAPTER

20

Automating Dreamweaver

One of Dreamweaver's greatest selling points is that it makes you more productive. You've experienced this firsthand if you've ever labored over tables in an HTML text editor. What once took many keystrokes now takes one click of the Insert bar's Table object.

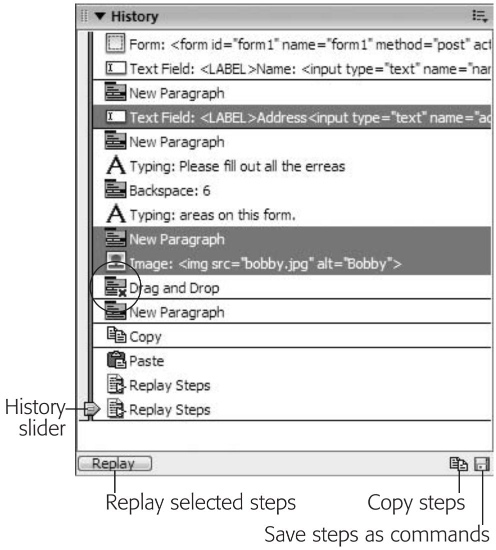
If you're looking for even more ways to cut time off your day, Dreamweaver doesn't disappoint. In addition to its Snippets, Library, and Template features (see Chapters 18 and 19), the program offers two tools that let you automate common and time-consuming tasks: the History panel and the Find/Replace command.

The History Panel Revisited

As you work on a Web page, Dreamweaver keeps track of everything you do. You can see a list of your actions—your *history*—in the History panel. Each document has a separate history, which Dreamweaver discards when you close the document or quit the program.

You can use the History panel to quickly undo or redo multiple actions (see page 81), but that's only the tip of the iceberg. You can also use it to replay and record a series of actions you wish to repeat. If you've ever used macros in Microsoft Word or actions in Adobe Photoshop, you'll probably get the hang of this feature quickly.

To open the History panel, choose Window → History, or press Shift+F10 (see Figure 20-1).



History

Form: <form id="form1" name="form1" method="post" act

Text Field: <LABEL>Name: <input type="text" name="nar

New Paragraph

Text Field: <LABEL>Address<input type="text" name="ad

New Paragraph

Typing: Please fill out all the erreas

Backspace: 6

Typing: areas on this form.

New Paragraph

Image: <img src="bobby.jpg" alt="Bobby">

Drag and Drop

New Paragraph

Copy

Paste

Replay Steps

Replay Steps

Figure 20-1:
The History panel lists every little step you've taken while working on the current document–even typos. You can replay one or more actions on the list, copy them for use in another document, or save them as a command in the Commands menu. If Dreamweaver can't replay an action, such as a mouse click, it appears with a red X next to it (circled). Furthermore, you can't replay two consecutive steps if you clicked or dragged in the document in between them (you'll see a solid line in the History list separating such steps). Dreamweaver merely replays the first selected step. The History slider indicates where you are in the document's history.

Replay Your Steps

To replay a step in the History panel, click the step's name to highlight it. You can also select multiple steps by using one of these methods:

- To select a group of consecutive steps, drag over them. You can drag your cursor across either the labels or icons. Take special care not to move your cursor onto the History slider on the left edge of the window, as clicking there undoes or redoes steps (page 83).
- You can also select consecutive steps by holding down the Shift key as you click down the list.
- To select steps that aren't consecutive, Ctrl-click (⌘-click) only the ones you want. For example, say you hit Return, typed *hello*, and then inserted a horizontal rule. If you wanted to omit the step where you typed *hello*, you could Ctrl-click (⌘-click) the other two. Dreamweaver ignores unselected steps.

Now, when you click Replay (see Figure 20-1), Dreamweaver replays the selected steps in an encore performance. For example, if you insert an image using the Common tab of the Insert bar (see page 200), that's recorded as one step. You could then add that image to your page again, later, simply by clicking where you want the image to appear and then replaying the "insert image" step. Unfortunately, you can't reorder the steps; they always play from the top of the list to the bottom.

Once you've created a series of steps, you can reuse it. For example, say you format a paragraph as a bulleted list and apply a custom style to it. Once Dreamweaver records these steps in the History panel, you can select more text and replay those steps to format it the same way. Now imagine that, instead of a two-step process, you have a 10-step chore that involves not only keystrokes, but multiple visits to the Insert bar and Property inspector—you can begin to see the power of this feature.

Tip: You probably know that you can repeat your last action by pressing Ctrl+Y (⌘-Y) or choosing Edit → Repeat. For example, if you type the word *hello* in the document window, pressing Ctrl+Y (⌘-Y) will type the word *hello* again. Unless you're Jack Nicholson's character in *The Shining*, this feature may sound less than useful, but used in combination with the History panel's Replay feature, it can be a real time-saver. When you use the History panel to replay several steps, you'll notice the last item in the History list becomes Replay Steps. Dreamweaver treats the replaying of all these steps as a *single action*. Now if you press Ctrl+Y (⌘-Y), you replay all of the steps again.

Exceptions and Errors

Unfortunately, Dreamweaver can't record and play back everything. The exceptions generally involve making changes in certain dialog boxes or moving objects with the mouse. For example, you can't record tasks you perform in the Modify Page Properties dialog box. And you're left to your own devices when you want to click, drag, or drop a graphic in the document window.

On the other hand, not everything you do with the mouse is off-limits to the History panel. It can track most common tasks, like clicking the Insert bar, choosing a menu item, or clicking in the Property inspector to set a property. Also, you can avoid using many mouse movements by using the equivalent keystrokes, which Dreamweaver *can* record. (See the box "Keyboard to the Rescue" below.)

If you take a step, such as a mouse drag, that Dreamweaver *can't* replay, a red X appears next to it in the History panel. A line between two actions also indicates a step that can't be repeated. This problem usually arises when you've clicked in the document window (to deselect an image, for example). One way you can avoid these non-recordable actions is to get into the habit of deselecting an object in the document window by pressing the keyboard's arrow keys instead.

FREQUENTLY ASKED QUESTION

Keyboard to the Rescue

If Dreamweaver can't track mouse movements, how can I replay an action that involves selecting something?

It's easy to use the mouse to make selections and move items around the screen, but you can do much of the same with the humble keyboard. That's a good thing, because if you can type it, Dreamweaver can record it.

To move up one line, for instance, press the up arrow key; to move down a line, press the down arrow. You can move to the top or bottom of the document window with the Page Up and Page Down keys, or move to the beginning or end of a line by pressing Home or End. Press Shift while pressing the right or left arrow key to select the object or letter to the right or left of the insertion point. Add the Ctrl (⌘) key to that combination to select one word at a time. Unfortunately, Dreamweaver doesn't record the keystrokes you use for moving between table cells (Tab and Shift-Tab). However, there's a workaround: to move from one cell to the cell on its right, press End, followed by the right arrow key. To move to the cell to the left, press Home, followed by the left arrow key. Arrow keys not only move the cursor but are also a helpful way to deselect an object that's currently highlighted on the page. Best of all, the History panel can track all of these keystrokes.

(You don't have to memorize all of this. You can print out a complete list of keyboard shortcuts, as described on page 733.)

Copying and Pasting Actions

Each document has its own history. So if you work on one page and then switch to another, the History panel changes to reflect only the actions you performed on the new document. The biggest drawback of this quirk is that you can't immediately replicate a series of steps you've made in one document by replaying them in another document.

For example, while working on your home page, you might click the Date object in the Insert bar to insert the current date (see page 70), and then choose a format for the date in the dialog box. Then, say you want to place the date on another page using the same format. But when you switch to that page and click Replay on the History panel, your steps aren't there!

Fortunately, there's a workaround: ye olde copy/paste routine. Select the steps you want to copy (see page 714 for selection techniques), and then click the "Copy selected steps" button (see Figure 20-1) on the History panel. (The regular copy shortcut, Ctrl+C or ⌘-C, *doesn't* work in this situation.) Now switch to the new document, select an object (or click to place the insertion point), and then choose Edit → Paste or press Ctrl+V (⌘-V).

Dreamweaver responds by playing the copied steps.

Save Steps as Commands

It's quick and easy to replay and copy steps to automate repetitive tasks, but if you close the document or quit Dreamweaver, your recorded actions disappear—and with them, any chance you had of replaying them in the future. What if you come up with a great sequence of steps that you'd like to use over and over again?

POWER USERS' CLINIC

Copy (and Study) Actions

Dreamweaver is relatively easy to customize, because the objects that appear in the Insert bar, the behaviors available from the Behaviors panel, and even the Property inspector are all–behind the scenes–combinations of HTML pages and JavaScript programs. If you understand JavaScript, you can add your own commands, behaviors, and objects.

When learning JavaScript, however, you may need all the help you can get. The History panel's Copy Steps feature is a good place to start.

To study how Dreamweaver's built-in commands, behaviors, and objects have been programmed, copy one or more actions using the method that's described above. Switch to a text editor like Notepad or TextEdit (Word will work, too), and then choose Edit → Paste.

What you see is the JavaScript code that Dreamweaver uses to carry out those actions. You'll find out, for example, that while you perceive adding a new paragraph to your Web page as a matter of hitting Enter, to Dreamweaver it looks like this: *dw.getDocumentDOM().newBlock()*.

The History panel also has a secret shortcut that lets you view the JavaScript code for each step, right inside the panel: Ctrl+Shift+click (⌘-Shift-click) anywhere inside the History panel (Ctrl+Shift+click [⌘-Shift-click] again to return to the normal, human-readable description for each step).

The solution: Before it disappears forever, turn it into a *custom command*. That way, Dreamweaver adds your command to the bottom of the Commands menu, and you can choose it from there whenever you want.

To save steps as a command, select the steps you want to copy (see page 714 for selection techniques), and then click the Save Steps button (its icon looks like a little floppy disk) on the History panel.

The "Save as Command" dialog box pops open. (If you've selected steps that Dreamweaver can't replay, such as mouse movements, a warning appears. Click Yes to continue without those steps; the valid steps work just fine.) Type a short, descriptive name, and then click OK. Now take a look at the Commands menu—sure enough, your command now appears at the bottom.

To use your custom command, simply select its name from the Commands menu.

Tip: If you decide you want to delete your command or change its name, choose Commands → Edit Command List. In the dialog box that appears, click the command's name to select it. Type a new name or click Delete.

Recording Commands

You can also create a command by telling Dreamweaver to watch and record your actions. This time, Dreamweaver doesn't *let* you perform mouse movements while you're recording, so you can be sure recorded commands will play back properly.

To record a command, make sure the relevant Web page document is frontmost, and then choose Commands → Start Recording, or press Ctrl+Shift+X (⌘-Shift-X). The cursor turns into a cassette-tape icon to indicate the command is recording. Now's your chance to do whatever you want Dreamweaver to memorize. (If you try to use the mouse to move or select anything in the document window, Dreamweaver complains with a dialog box.)

When you're finished, choose Commands → Stop Recording, or press Ctrl+Shift+X (⌘-Shift-X). Your cursor returns to normal, and Dreamweaver saves the sequence as a command, which you can replay by choosing Commands → Play Recorded Command.

Note, however, that this command disappears when you quit Dreamweaver or record another command. (Dreamweaver can only save one recorded command at a time.) If you want to preserve it for posterity, you have to save it to the Commands menu, like this:

1. **Choose Commands → Play Recorded Command.**

   The History panel lists this action as Run Command.

2. **Click the Run Command step in the History panel.**

   The step is highlighted to indicate you've selected it.

3. **Click the Save Steps button (its icon looks like a little floppy disk).**

   The “Save as Command” dialog box appears.

4. **Type a name for the command, and then click OK.**

   Dreamweaver adds your new command to the Commands menu, where it’s ready for action in this or any future Dreamweaver session.

Find and Replace

You’ve probably encountered find-and-replace tools in word processing programs and even some graphics programs. As the name implies, the command finds a piece of text (*Webmaster*, for example) and then *replaces* it with something else (*Webmistress*). Like Microsoft Word, Dreamweaver can search and replace text in the body of your Web pages. But it also offers variations on this feature that enhance your ability to work within the tag-based world of HTML.

What’s more, Dreamweaver lets you find and replace text on *every* page of your Web site, not just the current, open document. In addition, you can *remove* every appearance of a particular HTML tag, or search and replace text that matches very specific criteria. For example, you can find every instance of the word “Aardvark” that appears within a center-aligned paragraph. These advanced find-and-replace options are some of the most powerful—and underappreciated—tools in Dreamweaver’s toolbox. If you learn how to use them, you can make changes to your pages in a fraction of the time it would take using other methods.

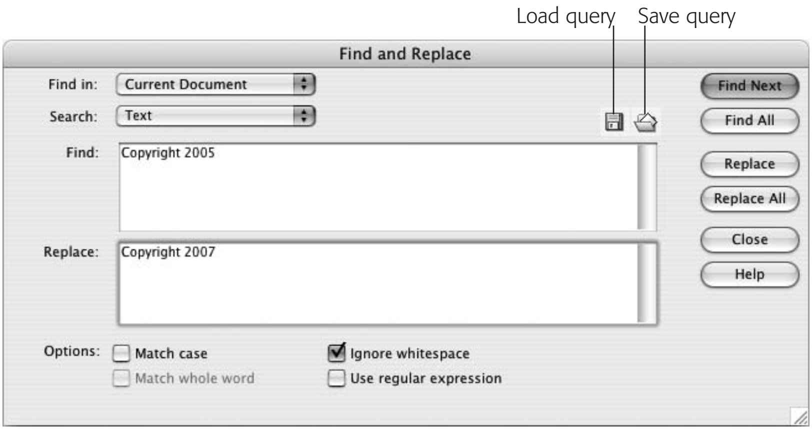
Tip: You can use “Find and Replace” to search an entire site’s worth of files. This is powerful, but can also be slow, especially if some folders hold files you don’t want to search–old archives, for example. You can use Dreamweaver’s cloaking feature to hide files from find-and-replace operations. See page 626 for more details.

Find and Replace Basics

To start a search, press Ctrl+F (⌘-F), or choose Edit → “Find and Replace.” The “Find and Replace” window opens (see Figure 20-2). Now all you have to do is fill in the blanks and set up the search.

Whether you perform a simple text search or a complex, tag-based search and replace, the procedure for using the “Find and Replace” command is basically the same. First, you need to tell Dreamweaver *where* to search (within text you’ve selected on the page, in a file, a folder, or an entire Web site). Next, tell it *what* to search for (text, HTML, or a particular tag with a specific attribute). Finally, you can decide what to replace the item with. This last step is optional; you can use the “Find and Replace” window as a way to locate an item on the page, or in your site, without actually changing it to anything.

Tip: After you've entered the "Find and Replace" criteria, click the Save Query button (see Figure 20-2). A Save dialog box appears; you can type in a name for your query, which Dreamweaver saves as a .dwr (Dreamweaver replace query) file. You can save this file anywhere on your computer. If it's a query you'll use for a particular site, you might want to save it with those files. To reuse a query, click the Load Query button and locate the .dwr file. After the search-and-replace criteria load, you can click any of the four action buttons–Find Next, Find All, Replace, or Replace All.



Load query Save query

Find in: Current Document

Search: Text

Find: Copyright 2005

Replace: Copyright 2007

Options: ☐ Match case ☑ Ignore whitespace

☐ Match whole word ☐ Use regular expression

Figure 20-2: *Dreamweaver's "Find and Replace" feature lets you replace text and HTML quickly and accurately. By using the Load Query and Save Query buttons, you can even save complex searches to use in the future.*

Basic Text and HTML Searches

Dreamweaver can either search all of the source code in a page or simply focus on text that appears in the document window.

- **A source code** search lets you find and replace any character in the code of a page, including words, letters, and symbols. This means *anything* you see in Code view, such as HTML, CSS, or server-side programming code used to create the dynamic database-driven sites described in Part 6 of this book.
- **Text** searches are more refined. They look only for text that appears within the body of a page. That is, Dreamweaver ignores HTML tags, properties, and comments when searching—in short, it ignores anything that doesn't appear as actual words in the document window. By using a text search when you want to change the word "center" to "middle," for example, you won't accidentally alter the center *alignment* option of a table cell—<td align="center">—by setting the alignment value of its HTML tag to "middle" (an invalid value that Web browsers would just ignore).

If you've used "Find and Replace" in other programs, the following routine will be familiar.

Phase 1: Determine the scope of your search

Using the "Find in" menu (see Figure 20-3), choose any of these options:

- **Selected Text.** Searches only the current selection of the Web page you're working on. This can be useful if you're working in Code view and you want to search the code in just a certain section of the page, such as the head of the document. It's also great when you're writing your own server-side programs (like those described in Part 6 of this book) and you want to search only one part of the code.
- **Current Document.** Searches the Web page you're working on.
- **Open Documents.** Searches all currently open Dreamweaver documents. This option is handy if you're working on a bunch of pages at the same time and realize you've made the same typo on each page.
- **Folder.** Search all Web pages in a particular folder. Dreamweaver also searches Web pages in all folders *within* the selected folder. You can use this option to search pages that aren't part of the current site.
- **Selected Files in Site.** To use this option, open the Site window and select files that you want to search in the local file list. (See page 588 for details.)
- **Entire Current Local Site.** Searches every Web page in the current site folder, including pages in folders *inside* the site folder. This option is invaluable when some basic piece of information changes throughout your site. For instance, use this when your boss's sex-change operation requires you to replace every instance of "Mark Jones" with "Mary Jones" throughout your company's site.

Warning: Using the "Find and Replace" command is one of the best ways to quickly make changes to an entire site, but it's also one of the easiest ways to wreck a site's worth of Web pages. Dreamweaver can't undo changes made by the "Find and Replace" command to files that aren't open on your computer. So be careful. If you plan on making extensive changes, make a backup copy of your files first!

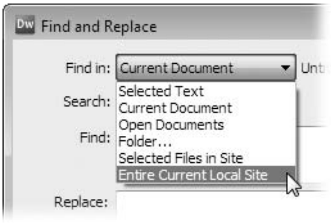


Figure 20-3:
The "Find and Replace" command is not limited to the current document. You can also search multiple Web pages, or even an entire site.

Phase 2: Specify what to search for

For your next trick, you'll tell Dreamweaver what you want to search for. Use the Search pop-up menu to choose one of these two options:

- **Text.** This makes Dreamweaver search for a certain word or phrase of text that appears in the *body* of the documents you've specified. Type the text you want to find into the Search field. If you're searching for a pattern in your text, enter a *regular expression* here and turn on the "Use regular expression" checkbox. (See the box on page 725 for more on regular expressions.)
- **Source Code.** Basic text searches are very useful, but they're limited to text that appears in the body of the page (what you see in the document window). If you want to search and replace code, you need the Source Code option.

  Source-code searches work identically to text searches, except that Dreamweaver searches *everything* within the file—text, HTML, JavaScript, CSS, and so on—and replaces any part of the file. Using this option, you could search for any instance of the tag <img src="mark_jones_photo.jpg">, for example, and then replace it with <img src="mary_jones_photo.jpg">.

  (If you're in Code view, Dreamweaver automatically selects the Source Code option.)

  As you fill in the Search field, be aware that some plain English words are also special words in HTML, JavaScript, or CSS. If you try to replace *table* with *desk* using a source-code find and replace, you'll completely destroy any <table> tags on the page.

  You can also enter a regular expression to search for patterns in your HTML source code (see the box on page 725).

Phase 3: Provide the replacement text

If you want to change the text that Dreamweaver finds, type the replacement text into the Replace box. It may be the word or words you'd like to swap in (for a Text search), or actual HTML code if you're performing a source-code search.

Tip: Dreamweaver won't let you create a new line in the search or replace boxes–for example, if you want to replace some source code with two lines of HTML. At least it won't let you do it the normal way. If you hit the Enter key, Dreamweaver begins the search rather than inserting a new line. If you want to add another line use Shift-Enter (Shift-Return).

If you just want to find the text without replacing it, then skip this step.

Tip: If you want to find the specified text and replace it with *nothing* (that is, deleting every occurrence of the text), leave the Replace field blank and perform a replace operation, described in Phase 5.

Phase 4: Choose the search settings

Dreamweaver gives you three options that govern its search and replace; some of them are quite complex:

- The **Match Case** option limits the Find command to text that exactly matches the capitalization you use in the Search field. If you search for the text *The End* with the Match Case box turned on, Dreamweaver finds a match in "The End is near," but not in "You're almost at the end." Use this trick to find every instance of *web* and replace it with *Web*.
- **Match Whole Word** searches for an entire word—not a portion of a larger word. For example, if you turn this option on, a search for *Rob* matches only "Rob," and not any parts of "Robert," "robbery," or "problem." If you don't select this option, Dreamweaver stops on "rob" in all four instances, and could cause serious pboblems if you also *replace* "rob" with something like "bob". (Note that if you selected the Match Case option, Dreamweaver would match *Rob* in "Rob" and "Robert," but *not* in "robbery" and "problem," since they don't include a capital R.)
- The **Ignore Whitespace** option treats multiple spaces, tabs, non-breaking spaces, and carriage returns as single spaces when searching. For instance, if you search for *the dog* and turn on this option, Dreamweaver matches "the   dog" as well as "the dog"—even if the multiple spaces are actually the HTML non-breaking space character *&nbsp;* (see page 68).

  Unless you have a good reason, always leave this option turned on. The HTML of a page can contain lots of extra spaces, line breaks, and tabs that don't appear in a Web browser or in Dreamweaver's document window. For example, in the HTML of a document, it's possible to have two lines of code that look like this:

  ```
  <p>This sentence will appear on one
  line in a Web browser</p>
  ```

  Even though this text would appear on a single line in the document window, a search for "one line" *without* the Ignore Whitespace box turned on would find no match. The carriage return after "one" is not an exact match for the space character in "one line."

Note: The Ignore Whitespace option can't be selected when the Use Regular Expression checkbox is turned on.

- The **Use Regular Expression** option is used for matching patterns in text. For a discussion of this advanced technique, see the box on page 725.

Phase 5: Take action

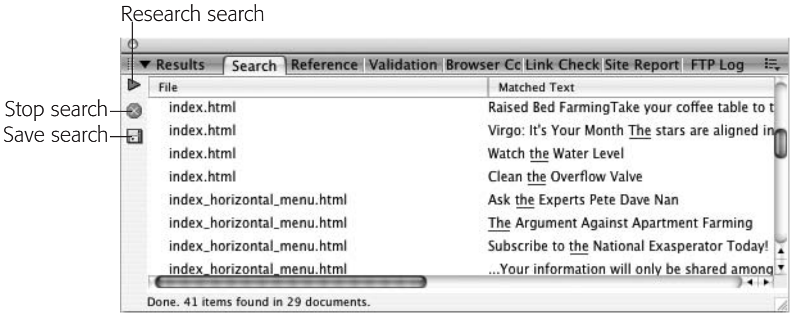
Finally, you're ready to set the search in motion by clicking one of the four action buttons in the "Find and Replace" window (see Figure 20-2):

- **Find Next** locates the next instance of the search term. If you're searching the current document, Dreamweaver highlights the matching text. If you're searching an entire Web site or a folder of pages, Dreamweaver opens the file *and* highlights the match. You can cycle through each instance of the search term by clicking this button repeatedly.

Tip: As in other programs (notably Microsoft Word), you can press Enter or Return to repeat the Find Next function. If you've clicked into the document window–or even closed the Find window–you can press F3 (⌘-G) to repeat the Find Next function.

- **Find All** locates every instance of the search terms, all at once, and shows them to you in a list in the Search tab of the Results panel (Figure 20-4). The name and location of each file (if multiple files are searched) appear to the left, and the matched text appears to the right. Dreamweaver displays part of the sentence in which the matched word or words appear. The exact match is underlined with a squiggly red line, so you can see the search in context and identify text you may *not* want to replace.

  Unlike the Find Next action, Find All doesn't automatically open any of the Web pages containing matches. Instead, to open a matched page, double-click its name in the results list. Only then does Dreamweaver open the Web page and highlight the match.



File	Matched Text
index.html	Raised Bed FarmingTake your coffee table to t
index.html	Virgo: It's Your Month The stars are aligned in
index.html	Watch the Water Level
index.html	Clean the Overflow Valve
index_horizontal_menu.html	Ask the Experts Pete Dave Nan
index_horizontal_menu.html	The Argument Against Apartment Farming
index_horizontal_menu.html	Subscribe to the National Exasperator Today!
index_horizontal_menu.html	...Your information will only be shared among

Done. 41 items found in 29 documents.

Figure 20-4: *The green-arrow button reopens the "Find and Replace" window. Click the red Stop button to abort the current search (for example, when you inadvertently begin a search for "the"). You can also save a rather useless XML file that provides a report of the results of the find-and-replace command (remember the old adage: Just because you can doesn't mean you should).*

- **Replace** locates the next instance of the search term *and* replaces it with the text in the Replace field, leaving the replaced text highlighted for your inspection.

You can use this button in combination with Find Next to selectively replace text. First, click Find Next. Dreamweaver locates and highlights the next match. If you want to replace the text, click Replace. If not, click Find Next to search for the next match, and repeat the cycle. This cautious approach lets you supervise the replacement process and avoid making changes you didn't intend.

- **Replace All** is the ultimate power tool. It finds every instance of the search term and replaces it with the text entered in the Replace field. Coupled with the "Find in Entire Local Site" option, you can quickly make sitewide changes (and mistakes—so back up all your files before you Replace All!).

  When you click this button, Dreamweaver warns that you can't undo this operation on any closed files. You can erase mistakes you make with the "Find and Replace" in *open* documents, by choosing Edit → Undo, but Dreamweaver *permanently* alters closed files that you search and replace. So be careful! (On the other hand, changes to open documents aren't permanent until you save those files.)

Tip: Before you take the plunge and click the Replace All button, it's a good precautionary step to click Find All first. This way you can be sure that you're going to change exactly what you *want* to change.

If you use the Find All or Replace All commands, the "Find and Replace" window closes, and the results of the search appear in the Search tab (see Figure 20-4). You can reopen the "Find and Replace" window (with all of your previous search criteria still in place) by clicking the green arrow on the Search tab—called the "Find and Replace" button—but only if you haven't selected anything else—like text on a page—first.

Advanced Text Searches

If you want greater control over a text search, you can use the "Find and Replace" command's *advanced* text search option, which lets you confine a search to text either inside or outside a specific tag.

For example, when Dreamweaver creates a new blank document, it sets the page's *Title* property to *Untitled Document*. Unfortunately, if you forget to change it, a site can quickly fill up with untitled Web pages. A basic text search doesn't identify this problem, because it searches only the body of a page; titles appear in the head. And a source-code search for *Untitled Document* would turn up the words "untitled document" *wherever* they appeared in the page, not just inside the <title> tag.

In cases like this, an advanced text search is your best choice. Simply set the "Find and Replace" command to search for *Untitled Document* whenever it appears within the <title> tag.

To use the advanced text search, you follow the same general routine as described on the previous pages. But before using one of the action buttons, you make a few additional setup changes to the dialog box.

POWER USERS' CLINIC

Turbocharge Your Searches

If you want to find the phone number 555-123-5473, no problem; just type *555-123-5473* into the search field. But what if you wanted to find any and every phone number–555-987-0938, 555-102-8870, and so on–on a Web page or in a site?

In such a case, you need to use *regular expressions*, the geeky name for a delightfully flexible searching language, carried over from early UNIX days, which consists of wildcard characters that let you search for patterns of text instead of actual letters or numbers. Each phone number above follows a simple pattern: three numbers, a dash, three more numbers, another dash, and four more numbers.

To search for a pattern, you use a variety of symbols combined with regular text characters to tell Dreamweaver what to find. For example, in the world of regular expressions, \d stands for "any number." To find three numbers in a row, you could search for \d\d\d, which would find 555, 747, 007, and so on. There's even shorthand for this: \d{3}. The number between the braces ({}) indicates how many times in a row the preceding character must appear to match. To finish up the example of the phone numbers, you could use a regular expression like this: \d{3}-\d{3}-\d{4}. The \d{3} finds three numbers, while the hyphen (-) following it is just the hyphen in the phone number, and \d{4} finds four numbers.

Here are some of the other symbols you'll encounter when using regular expressions:

- **.** (period) stands for any character, letter, number, space, and so on.
- **\w** stands for any letter or number.
- ***** (asterisk) represents the preceding character, zero or more times (and is always used after another character). This is best explained with an example: The regular expression colou*r, for instance, matches both colour and color–the * following the u indicates that the u is optional (it can appear zero times). This would also match colouuuuur (handy for those times when you've fallen asleep on the keyboard).

To see a complete list of the regular-expression characters Dreamweaver understands, launch the Dreamweaver online Help system (press F1) and search for the topic "Regular Expressions." A full-length discussion of regular expressions could–and does–fill a book of its own; check out *Mastering Regular Expressions* (O'Reilly) by Jeffrey E. F. Friedl or, for made-to-order regular expressions, check out *Regular Expression Recipes* (Apress) by Nathan Good.

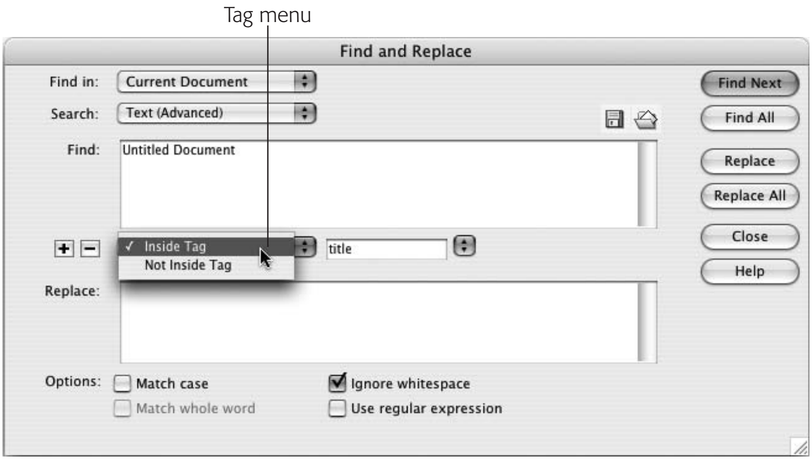
For an example of using regular expressions in Dreamweaver, see page 730.

Limiting the search by tag

Choose Text (Advanced) from the Search pop-up menu to make the expanded controls appear (see Figure 20-5).

Now, from the menu next to the + and – buttons, choose either Inside Tag or Not Inside Tag. For example, consider this line of code: "Stupid is as <strong>stupid </strong> does." The first instance of "stupid" isn't inside the <strong> tag, but the second one is.

Note: A more descriptive name for the first option would be *"Enclosed* By Tag"; Dreamweaver actually searches for text that's between *opening and closing* tags. In fact, an advanced text search using this option doesn't identify text that's literally inside a tag. For example, it won't find "aliens" in this line of code: <img src="ufo.jpg" alt="Aliens live among us.">, but it would find "aliens" in this one: <strong> Aliens live among us.</strong>. In the first example, *aliens* appears as part of the <img> tag, while in the second, *aliens* is enclosed by the opening and closing <strong> tags.



Find in: Current Document

Search: Text (Advanced)

Find: Untitled Document

✓ Inside Tag title

Replace:

Options: □ Match case ☑ Ignore whitespace

□ Match whole word □ Use regular expression

Figure 20-5: *Use an advanced text search to limit your search to text that appears within a particular HTML tag. Or, conversely, use it to search for text that doesn't appear in a particular tag.*

Once you've specified whether you're looking for text inside or outside tags, you can choose a specific HTML tag from the Tag menu identified in Figure 20-5.

The menu lists all HTML tags—not just those with both an opening and closing tag. So the image tag (<img>) still appears, even though Dreamweaver doesn't identify text inside it.

Tip: A great way to search for text in both the title and body of a Web page is to choose the Inside Tag option and then select *html* from the Tag menu. That way, you can search for any text that appears within the opening <html> and closing </html> of the page–which, since those tags start and end any Web document, is *all* text on a page. This trick is handy when you want to change text that might appear in the body *and* the title of a page (for example, a company name).

Limiting the search by attribute

To limit the search further, click the + button (see Figure 20-6); yet another set of fields appears.

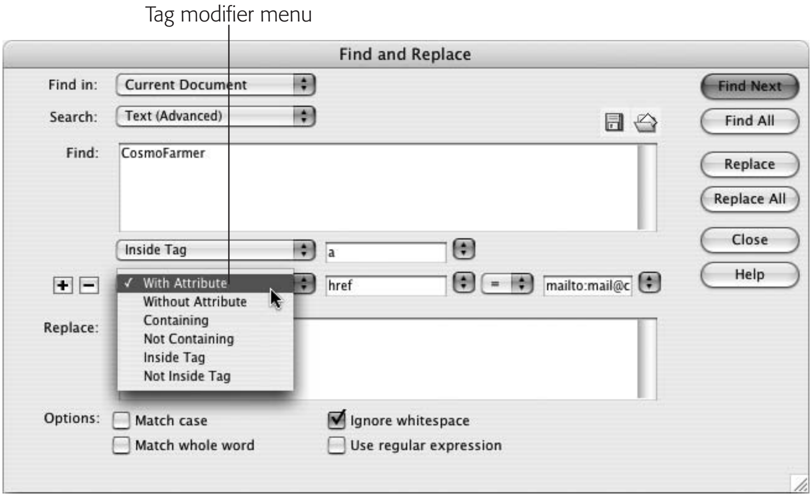
Using the Tag Modifier menu—next to the + and – buttons (Figure 20-6)—you can choose from any of six options that break down into three groups:

- **With Attribute/Without Attribute.** To limit the search, you can specify that a tag must either have (With Attribute) or not have (Without Attribute) a specific property.

  For example, say the following lines of code appear throughout a Web site:

  ```
  <p>For assistance, please email
  <a href="mailto:mail@cosmofarmer.com">
  CosmoFarmer.</a></p>
  ```

  Now, for the sake of argument, say you need to change it to read "For assistance, please email Customer Service." A basic text find-and-replace would



Tag modifier menu

Find and Replace

Find in: Current Document

Search: Text (Advanced)

Find: CosmoFarmer

Inside Tag a

With Attribute href = mailto:mail@c

Options: □ Match case ☑ Ignore whitespace □ Match whole word □ Use regular expression

Figure 20-6: *When you click the + button in the "Find and Replace" window, a new set of fields appears. Use these options to carefully hone your "Find and Replace" commands, and zero in on text that matches precise criteria.*

incorrectly change the word "CosmoFarmer" to "Customer Service" *everywhere* on the site.

However, an advanced text search using the With Attribute option would let you specifically target the text "CosmoFarmer" wherever it appears inside an <a> tag whose *Href* attribute is set to *mailto:mail@cosmofarmer.com*. You could then just change that text to "Customer Service" while leaving all other instances of "CosmoFarmer" alone. (To learn about the different HTML tags and attributes, use Dreamweaver's built-in code reference; see page 394.)

After you choose With Attribute, use the menu on the right to select *which* of the tag's properties you want to find. (Dreamweaver automatically lists properties that are appropriate for the tag you've specified.) For example, if you search inside a <table> tag, the menu lists such properties as *align, background, bgcolor*, and so on.

Advance to the next pop-up menu to choose a type of comparison: = (equal to), != (not equal to), > (greater than), or < (less than). These options are useful only when the property's value is a numeric amount, such as the *Width* property of a table cell. In this way, you could locate all table cells that are wider than 100 pixels (width > 100). (This setting has no effect on values that are words, such as *center* in this example: <td align="center">.)

Finally, type the value of the property in the last field. If you were searching for a black-colored background, the value would be *#000000* (the hex value for black; see page 47 for more on working with colors).

You can also click the menu and choose "[any value]"—a useful option when you want to find tags that have a certain property, but you're not interested in the property's value. For example, if you want to find all <table> tags with a background color (no matter whether the color's #336699, #000000, or #FFFFFF), choose the *Bgcolor* attribute and "[any value]."

- **Containing/Not Containing.** These options let you specify whether the tag contains (or does not contain) specific text or a particular tag.

  When you choose this option, a different set of fields appears. Choose either Text or Specific Tag from the menu to the right, and then either enter some text or select a tag in the last field in the row.

  For example, another solution to the problem above would be to search for the text "CosmoFarmer" wherever it appears inside a <p> (paragraph) tag that *also contains* the text "please email."

- **Inside Tag/Not Inside Tag.** These last two choices are identical to those described on "Limiting the search by tag". They let you specify whether the tag is inside—or not inside—a specific tag. Use these to limit a search, for example, to text that appears only within a <span> tag that's *inside* a Heading 1 (<h1>) tag.

If you like, you can add even more restrictions to your search, adding new rules by clicking the + button and repeating the setup just described. When you're really on a roll, it's even possible to add so many modifiers that the "Find and Replace" window actually grows past the bottom of your monitor. To remove a modifier, click the minus sign (–) button.

Advanced Tag Searches

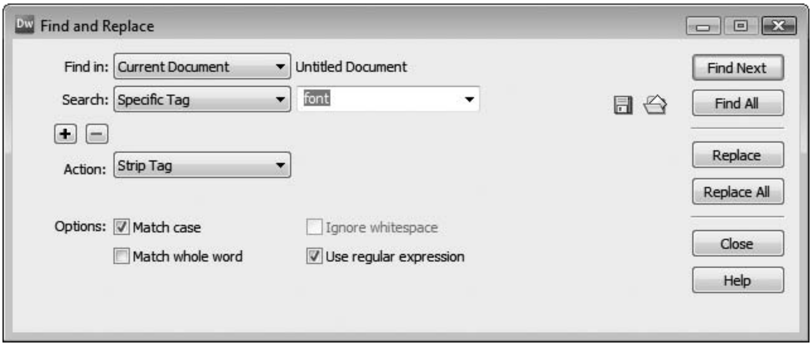
If you find the number of options an advanced text search offers overwhelming, you haven't seen anything yet. Dreamweaver's tag search adds more choices to help you quickly search for, and modify, HTML tags. You can use a tag search to strip out unwanted HTML tags (for example, if you're migrating an old site to CSS, you could remove the <font> tag), transform one tag into another (you could turn old-style *bold* [<b>] into the more widely accepted *strong* [<strong>] tag), and perform a host of other powerful actions.

In its basic outline, a tag search is much like the regular text search described on page 719. But this time, from the Search menu, you should choose Specific Tag.

Now a Tag menu appears next to the Search menu, and the dialog box expands to display a new set of fields (see Figure 20-7). Some of them are the same as the controls you see when performing an advanced text search (page 724), such as the Tag Modifier menu and the + button that lets you add additional restrictions to the search.

But a key difference here is the Action menu (Figure 20-8), which lets you specify the action Dreamweaver will perform on tags that match the search criteria when you click Replace or Replace All (if you intend to search, but not replace, then these options don't apply):

- **Replace Tag & Contents** replaces the tag, and anything enclosed by the tag (including other tags), with whatever you put into the With box to the right of this menu. You can either type or paste text or HTML here.



Find in: Current Document — Untitled Document

Search: Specific Tag — font

Action: Strip Tag

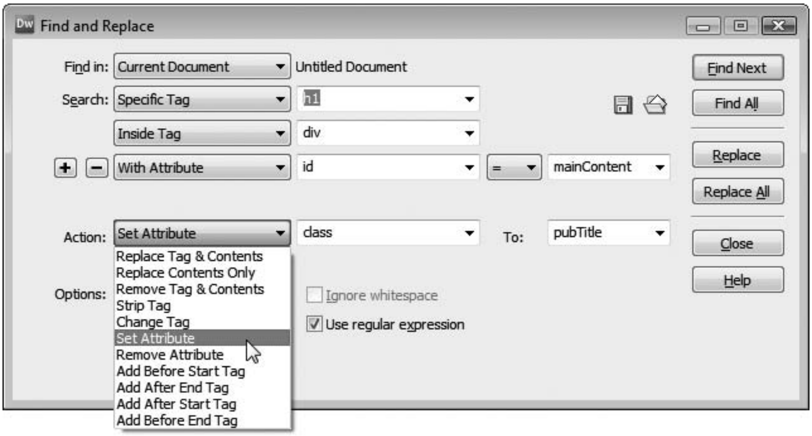
Options: ☑ Match case ☐ Match whole word ☐ Ignore whitespace ☑ Use regular expression

Figure 20-7:
It's a snap to remove tags when you use the Specific Tag option in Dreamweaver's "Find and Replace" command—just select the Strip Tag action. This option is handy if you're replacing old-style text formatting with Cascading Style Sheets. Use it to strip out unwanted <font> tags, for example.

- **Replace Contents Only** replaces everything enclosed by the tag with text or HTML that you specify. The tag itself remains untouched.

Note: Depending on which tag you're searching for, you might not see all the actions listed here. For example, the <img> tag doesn't have both an opening and closing tag like the <p> tag, which surrounds text inside a paragraph, so you won't see any of the options such as "Replace Contents Only" that affect the content between an opening and closing tag.

- **Remove Tag and Contents** deletes the tag and *everything* inside.
- **Strip Tag** deletes the tag from the page, but leaves anything enclosed by the tag untouched. The outmoded <font> tag is a perfect candidate for this action.
- **Set Attribute** adds an attribute to the tag. For example, you could set the *Alt* property of an image this way (see the example in the next section).
- **Remove Attribute** removes an attribute from a tag. For example, you could remove the not-so-useful *Lowsrc* attribute from all image tags on your pages.



Find in: Current Document — Untitled Document

Search: Specific Tag — h1

Inside Tag — div

With Attribute — id = mainContent

Action: Set Attribute — class To: pubTitle

Options: ☐ Ignore whitespace ☑ Use regular expression

Figure 20-8:
Once Dreamweaver finds a specific tag, it can perform any of 11 different actions on the tag or its contents.

- **Add Before (After) Start (End) Tag.** The last four actions in the menu simply offer variations on the same theme. They each let you place content in a Web page just before or after the tag for which you're searching.

  To understand how these actions work, remember that most HTML tags come in pairs. The paragraph tag, for example, has an opening tag (<p>) and a closing tag (</p>). Say you searched for a paragraph tag; you could add text or HTML *before* or *after* the start tag (<p>), or *before* or *after* the end tag (</p>). (For an example of this action at work, see the box on page 732.)

A Powerful Example: Adding Alt Text Fast

You've just put the finishing touches on the last page of your brand-new, 1,000-page site. You sit back and smile—and then snap bolt upright when you notice you forgot to add an Alt description for the site's banner graphic (see page 209). This graphic, called *site_banner.gif*, appears on every single one of the 1,000 pages. With rising dread, you realize that you'll have to open each page, select the graphic, and add the *Alt* property by hand.

And then you remember Dreamweaver's advanced tag-based find-and-replace feature.

Here's what you do. Press Ctrl+F (⌘-F) to open the "Find and Replace" window. Set up the dialog box like this:

1. **From the "Find in" menu, choose Entire Current Local Site.**

   You want to fix *every* page on your site.

2. **From the Search pop-up menu, choose Specific Tag; from the pop-up menu to its right, choose "img."**

   You'll start by identifying every image (the <img> tag).

3. **On the next row, use the three pop-up menus to choose With Attribute, "src," and the equals sign (=).**

   This tells Dreamweaver to look for specific images—in this case, images with a *Src* attribute (the path that tells a Web browser where to find the image file on the Web server) with a specific value.

4. **Type *.*site_banner\.gif* in the box next to the = sign.**

   For this exercise, assume the graphic file is stored in a folder called *images* located in the root folder of the site. The name *site_banner.gif* is the name of the image file. The .* is the magic, and you'll learn its purpose in a moment (ditto the backslash hanging out before the second period).

5. **Click the + button.**

   Another row of Tag Modifier menus appears.

6. **From the new row of menus, choose Without Attribute and "alt".**

   You've further limited Dreamweaver's search to only those images that don't already have the *Alt* attribute. (After all, why bother setting the *Alt* property on an image that already has it?)

7. **From the Action menu, choose Set Attribute; from the Tag menu, choose "alt".**

   You've just told Dreamweaver what to do when you click the Replace or Replace All button. When Dreamweaver finds an <img> tag that matches the search criteria, it will then *add* an *Alt* property to that tag.

   In this example, you might type *CosmoFarmer* in the To field; you've just specified the Alt text for Dreamweaver to add to the image.

8. **Turn on "Use regular expressions".**

   Regular expressions, described on page 725, let you search for specific patterns of characters and, in this case, help you accurately identify the banner graphic file everywhere it appears.

   You know you're looking for the file *site_banner.gif* wherever it appears in the site. Unfortunately, if you just type *site_banner.gif* as the value of the *Src* property of step 3, Dreamweaver can't succeed in its task. That's because the *Src* attribute, the part of the <img> tag that includes the name of the file, varies from page to page. Depending on where a page is relative to the graphic, the Src might be *site_banner.gif, images/site_banner.gif*, or even *../../../images/site_banner.gif*. What you need is a way to match every *Src* attribute that *ends* in *site_banner.gif*.

   A simple regular expression, *.*site_banner\.gif*, does the trick. The period stands for *any* character (6, g, or even %, for example), while the * (asterisk) means "zero or more times." When you add these codes to the graphic name, *site_banner.gif*, you instruct Dreamweaver to find every *src* value that ends in *site_banner.gif*.

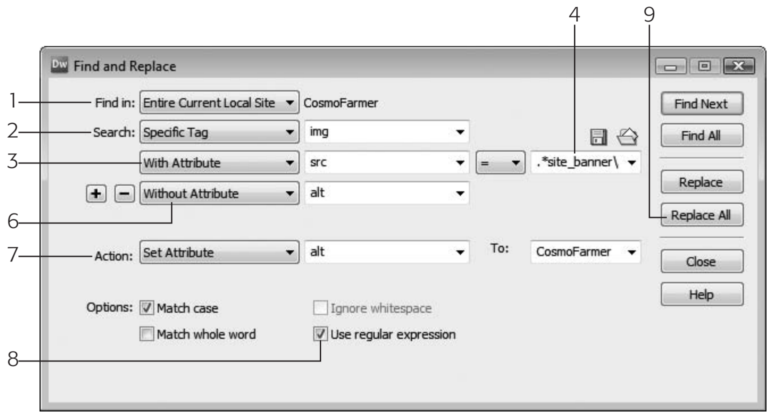
   Note the backslash before the last period: *\.gif*. Since in the world of regular expressions, a period means "any character," simply using *site_banner.gif* would not only match *site_banner.gif*, but also *site_banner1gif, site_bannerZgif*, and so on—in other words, any character between *site_banner* and *gif*. The backslash means treat the next character literally; it's just a period with no special regular-expression power.

   The dialog box should look like the one in Figure 20-9.

9. **Click the Replace All button and sit back.**

   In a matter of moments, Dreamweaver updates all 1,000 pages.

   To test this out first, you might try a more cautious approach: Click the Find Next button to locate the first instance of the missing *Alt* property; verify that it's correct by looking in the Search box (see Figure 20-4); and then click the Replace button to add the proper Alt value. Double-check the newly updated



Find and Replace

Find in: Entire Current Local Site CosmoFarmer

Search: Specific Tag img

With Attribute src = .*site_banner\

Without Attribute alt

Action: Set Attribute alt To: CosmoFarmer

Options: ☑ Match case □ Ignore whitespace

□ Match whole word ☑ Use regular expression

Figure 20-9: *The numbers shown here correspond to the steps in this fictional example, in which you want to add an <alt> tag—for the benefit of people who can't, or don't want to, see graphics in their browsers—to every occurrence of the banner logo.*

FREQUENTLY ASKED QUESTION

Convenient Copyright Notices

I want to add a copyright notice to the bottom of each page in my Web site. Is there a way to automate this process so I don't have to edit every page in my site by hand?

You bet. Use Dreamweaver's "Find and Replace" command to add text or HTML to the bottom of any Web page. The trick is knowing how to use the command's Specific Tag option.

First, choose Edit → "Find and Replace" to open the "Find and Replace" window. Next, choose Entire Current Local Site from the "Find in" menu, and choose Specific Tag from the Search menu. Choose "body" from the Tag menu. Remember, the <body> tag in HTML encloses everything that appears inside a browser window; it's equivalent to what you see in the document window.

From the Action menu, choose Add Before End Tag. The end tag in this case is </body>. Since </body> marks the end of any content in a Web page, whatever appears directly before this closing tag will appear at the bottom of the Web page (you can probably see where this is going).

Now, in the text field next to the Action menu, type (or paste) the copyright notice you'd like to use on each page. You may want to first design the copyright message using Dreamweaver, and then copy and paste the HTML into this field.

Click Replace All. Dreamweaver handles the rest.

page to make sure everything worked as planned. You can continue updating pages one at a time this way, or, once you're sure it works correctly, press Replace All.

CHAPTER 21

Customizing Dreamweaver

Whether you're a hard-core HTML jockey who prefers to be knee-deep in Code view, or a visually oriented, drag-and-drop type who never strays from the document window, Dreamweaver lets you work whichever way you want.

By now, you're probably already using the Favorites tab on the Insert bar to store your most frequently used objects in one place, as discussed on page 170. But don't stop there. Dreamweaver also gives you the power to add, change, and share keyboard shortcuts—a simple way to tailor the program to your needs. If that's not enough of an efficiency boost, you can add features that even Adobe's engineers never imagined, from simple productivity add-ons like Quick-Link (see page 166) to advanced Server Behaviors to help power a complete e-commerce Web site. Dreamweaver's design allows amateur and professional programmers alike to write new features and functions using HTML, JavaScript, and XML (Extensible Markup Language). There are hundreds of these extras, called *extensions*, for you to explore. Best of all, you can try many of them for free.

Keyboard Shortcuts

As you use Dreamweaver, you'll hit the same keyboard shortcuts and travel to the same palettes and menus time and again. Perhaps your site uses a lot of graphics and Flash movies, and you're constantly using the keyboard shortcuts to insert them. But you may find that, after the thousandth time, Ctrl+Alt+F (⌘-Option-F) hurts your pinkie and uses too many keys to be efficient. On the other hand, the things you do all the time—like inserting text fields into forms or adding rollover images, for instance—may not have shortcuts at all, so you're forced to go to a menu.

To speed up your work and save your tendons, Dreamweaver comes with a keyboard-shortcut editor that lets you define or redefine shortcuts for most of the program's commands.

Dreamweaver stores keyboard shortcuts in sets. It's easy to switch between them—a useful feature when you share your computer with someone who likes different keystrokes.

Four sets come with the program:

- **Dreamweaver Standard**. When you first fire up Dreamweaver, this is the set that's turned on. It's the same set of keyboard shortcuts available in Dreamweaver 8 and it matches the keyboard shortcuts found in Fireworks.
- **Dreamweaver MX 2004**. Some keyboard shortcuts have changed since Dreamweaver MX 2004—for example, Shift-F5 instead of Ctrl-F5 now opens the Tag Editor window. But the changes are so minor, it's not really necessary to use this set.
- **BBEdit**. If you're a Mac user with a code-editing past, you may have spent a lot of time learning shortcuts for Bare Bones Software's popular BBEdit. If so, choose this set.
- **HomeSite**. Likewise, if you're adept at the Windows HTML text editor HomeSite, you may want to use its keyboard shortcuts.

You can access the shortcut sets from the Keyboard Shortcuts window. Choose Edit → Keyboard Shortcuts. Be patient—the sets can take some time to load. Once the dialog box appears, you can switch sets by choosing a new one from the Current Set menu (see Figure 21-1).

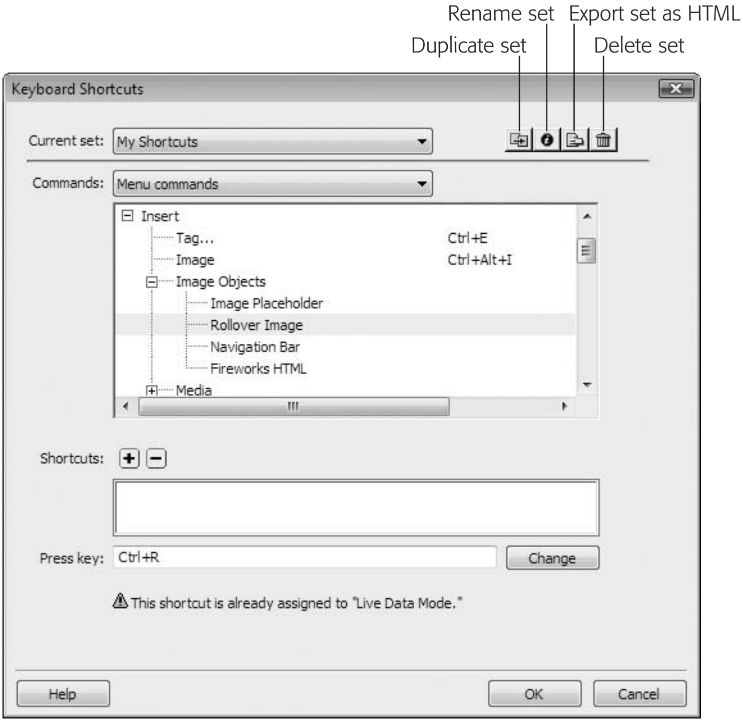
Make Your Own Shortcut Set

What if you want a set that *combines* BBEdit shortcuts with your most-used Dreamweaver ones? Or, you're a radical individualist who wants to remap *every* command to the keys of your liking? You can easily create keyboard shortcut sets that fit the way you work. Dreamweaver doesn't let you alter any of the four standard sets, so if you want to create your own, the first step is to make a copy of an existing one.

To do so, choose Edit → Keyboard Shortcuts (on the Mac, it's Dreamweaver → Keyboard Shortcuts). In the Keyboard Shortcuts window, use the Current Set pop-up menu to choose the set you wish to copy, and then click the Duplicate Set button (see Figure 21-1). Dreamweaver asks you to name the new set; do so, and then click OK.

You can delete or rename any set you create—once you figure out that the button in the Shortcuts window with the cryptic icon is the Rename Set button (see Figure 21-1). The Trash Can button, of course, lets you delete a set.

Tip: Dreamweaver lets you delete the four main keyboard shortcut sets. If you want one of them back, don't worry. The actual file isn't gone. You just need to edit a file called *mm_deleted_files.xml* in your Dreamweaver configuration folder. Remove the line that lists the keyboard shortcut set you want to get back and save the file. Then quit and restart Dreamweaver. (Note that each account holder on Windows and Mac OS X maintains a separate Configuration folder. See the box on page 744.)



Current set: My Shortcuts

Commands: Menu commands

```
⊟ Insert
    Tag...                      Ctrl+E
    Image                       Ctrl+Alt+I
  ⊟ Image Objects
      Image Placeholder
      Rollover Image
      Navigation Bar
      Fireworks HTML
  ⊞ Media
```

Press key: Ctrl+R

⚠ This shortcut is already assigned to "Live Data Mode."

Figure 21-1: *The Keyboard Shortcuts window lets you select or duplicate a shortcut set, as well as add and remove keyboard shortcuts for every menu item in Dreamweaver. You can also create keyboard shortcuts for Snippets (see page 647). When you attempt to create a shortcut that another command's already using, Dreamweaver warns you. If you wish, you can ignore the warning and reassign the keys to the new command.*

Changing Keyboard Shortcuts

Once you've created a new set, you can select any command and alter its shortcut. Start by choosing Edit → Keyboard Shortcuts (Dreamweaver → Keyboard Shortcuts) to open the Shortcuts window, if it's not already open. Then:

1. **From the Commands pop-up menu, choose the command type.**

   Dreamweaver organizes shortcuts into four (Macintosh) or seven (Windows) primary categories. These categories don't always make sense: For example, Copy and Paste appear under the Code Editing category, even though you use them at least as frequently while editing a document in the visual Design view. In addition, you'll find quite a few commands listed under multiple categories (you only need to change a keyboard shortcut once for these redundant listings).

Browse to see which commands have (or could have) keyboard shortcuts associated with them:

- **Menu commands** are the commands in Dreamweaver's menus, such as Insert → Image.
- You might presumably use the **Code editing** commands when editing HTML code. However, you could just as easily use them in Design view. They include Cut, Paste, and "Move to Top of Page," to name a few.
- **Document editing** commands are for selecting text and objects on a page, as well as previewing a page in a Web browser.
- **Files panel options menu** (Windows only) are the commands available when you right-click on a file in the Files panel.
- **Site panel** (Windows only) are the commands available from the contextual menu at the top right of the Files panel, such as Site → New Site. (On the Mac, many of these commands are listed in the Menu Commands category.)
- **Site window** (Windows only) are an odd assortment of commands that apply to situations like closing a window, quitting the application, or canceling FTP. (On the Mac, these commands are listed in the Document Editing group.)
- **Snippets** are reusable code pieces that you select from the Snippets panel, as discussed in Chapter 18.

2. **In the list below the Commands menu, click the command whose keyboard shortcut you want to change.**

   You'll find menu commands grouped by menu name: commands in the File menu, like Open and Save, fall under File. Click the + (Windows) or flippy triangle (Mac) next to the menu name to display the list of commands hidden underneath (see Figure 21-1).

   If the command already has a keyboard shortcut, it appears in the right-hand column. If it doesn't have a shortcut, you see an empty space.

3. **Click inside the "Press key" field, and then press the new keystroke.**

   Unless you're assigning the shortcut to an F-key or the Esc key, you must begin your shortcut with the Ctrl key (⌘-key). For example, the F8 key is a valid shortcut, but the letter R isn't; press Ctrl+R (⌘-R) instead.

Note: Some keyboard shortcuts may already be in use by your operating system, so assigning them in Dreamweaver may have no effect. For example, in Windows, Ctrl+Esc opens the Start Menu, while in Mac OS 10.4 (Tiger), Dashboard uses the F12 key.

Of course, many commands already have shortcuts. If you choose a key combination that's in use, Dreamweaver tells you which command has dibs. You can pick a different key combination, or simply click the Change button to reassign the shortcut to your command.

FREQUENTLY ASKED QUESTION

Sharing Shortcuts

How do I share my keyboard set with other people?

Dreamweaver stores your keyboard shortcuts as XML files—but in Dreamweaver finding them can be tricky. These files are in different locations depending on your operating system. Each keyboard set lives in an XML file; the file's name ends with the extension *.xml*.

In Windows XP, you'll find the custom keyboard set on your main hard drive in Documents and Settings → [Your Name] → Application Data → Adobe → Dreamweaver 9 → Configuration → Menus → Custom Sets. In Windows Vista, they're in *C:\Users\[your user name]\AppData\Roaming\Adobe\Dreamweaver 9\Configuration\Menus\Custom Sets.*

In Mac OS X, these files are squirreled away in your Home folder → Library → Application Support → Adobe → Dreamweaver 9 → Configuration → Menus → Custom Sets.

You can copy these files and place them in the Custom Sets folder on other computers. Once you've done so, Dreamweaver fans on those machines can use the Keyboard Shortcuts window (Edit → Keyboard Shortcuts or, on the Mac, Dreamweaver → Keyboard Shortcuts) to select the new set, just as though it had been created in that copy of Dreamweaver.

4. **Click the Change button.**

   Dreamweaver saves the new shortcut in your custom set.

   Repeat from step 1 if you want to make other keystroke reassignments. When you're finished, click OK to close the dialog box.

What if a command you use often doesn't have a shortcut at all? It's no problem to create one. As a matter of fact, Dreamweaver lets you assign *two* keyboard shortcuts to every command—one for you, and one for your left-handed spouse, for example.

To give a command an additional shortcut (or its first):

1. **Choose the command.**

   Follow the first two steps of the preceding instructions.

2. **Click the + button next to the word "Shortcuts".**

   The cursor automatically pops into the "Press key" field.

3. **Press the keys of your additional shortcut, and then click the Change button again.**

   Repeat from step 1 if you want to make other keystroke reassignments; when you're finished, click OK.

Deleting shortcuts is just as easy. Simply click the command in the list, and then click the minus sign (-) button next to the word "Shortcuts".

Create a Shortcut Cheat Sheet

Unless your brain is equipped with a 400-gig hard drive, you'll probably find it hard to remember all of Dreamweaver's keyboard shortcuts.

Fortunately, Dreamweaver offers a printable cheat sheet for your reference. At the top of the Shortcuts window, there's a handy "Export Set as HTML" button. (It's labeled with a cryptic icon; see Figure 21-1.) Click this button to name and save a simple HTML page that lists all of the commands and keyboard shortcuts for the currently selected set. Once you've saved the file, print it out or use it as an online reference—a great way to keep a record of your shortcuts for yourself or a team of Web page designers.

Dreamweaver Extensions

While keyboard shortcuts give you an easy way to access frequently used commands, they're not much help if the command you want doesn't exist. Suppose, for example, that you use Dreamweaver's Open Browser Window behavior (page 519) to load a new Web page into a window that's exactly 200 × 300 pixels. But what if you wanted the window to be centered in the middle of the visitor's montior? Dreamweaver's behavior doesn't do that. What's a Web designer to do? You could go to the Adobe site and request the new feature (*www.adobe.com/cfusion/mmform/index.cfm?name=wishform&product=12*) in hopes that the bustling team of programmers will add the command to the next version. But you'd have to wait—and there's no guarantee that Adobe would add it.

The legions of hard-core Dreamweaver fans have taken this feature wish-list issue into their own hands. As it turns out, amateur (and pro) programmers can enhance Dreamweaver relatively easily by writing new feature modules using the basic languages of the Web: HTML, JavaScript, and XML. (In fact, HTML forms, JavaScript programs, and XML documents constitute much of the program. The objects in the Insert bar, for example, are actually HTML pages stored within Dreamweaver's Configuration folder, and all of Dreamweaver's menus have actually been written as an XML file.)

Because of this "open architecture," you can add new functions and commands—called *extensions*—to Dreamweaver by downloading the work of all of those programmers. A Dreamweaver extension can take many forms and work in a variety of ways to change how the program works. It can be an icon on the Insert bar, a behavior listed on the Behaviors panel, or a command in the Commands menu. It might even be an entirely new floating window, like the Property inspector, that you use to alter some aspect of your page.

Best of all, whereas programming ability may be required to *create* extensions, none at all is necessary to use them. You can download hundreds of extensions from the Web and install them on your computer for free. In addition, many sophisticated extensions, like those for creating e-commerce sites, are commercially available.

Note: Extensions have been around for many versions of Dreamweaver. Unfortunately, each version of Dreamweaver added a few kinks for extension developers, so not all extensions out there work with Dreamweaver CS3. (Many extensions that were compatible with Dreamweaver 8 *do* work with Dreamweaver CS3.) Most extension developers list which versions of Dreamweaver their extensions work with, and you can also check for version compatibility on the Dreamweaver Exchange (see Figure 21-2).

Browse the Exchange

The largest collection of extensions awaits you at the Adobe Exchange Web site, where you'll find hundreds of free and commercial extensions. Although some come from Adobe itself, the vast majority are written by an army of talented Dreamweaver users.

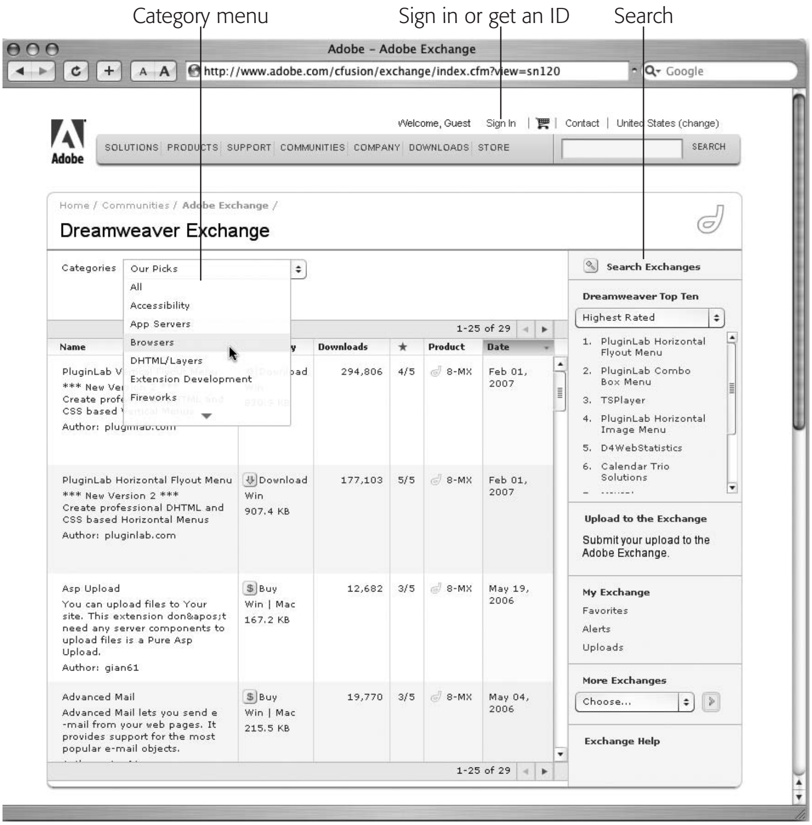
Using the Exchange is a straightforward process:

1. **In your Web browser, go to *www.adobe.com/exchange/dreamweaver*.**

   You can also get there from within Dreamweaver by choosing Commands → Get More Commands.

2. **Log in (see Figure 21-2).**

   You can *browse* the site without logging in, but to *download* any of the extensions, you need to get a free Adobe ID and sign in, using the Exchange Sign In form.



Category menu

Sign in or get an ID

Search

Submit your upload to the Adobe Exchange.

Name	…y	Downloads	★	Product	Date
PluginLab V… *** New Ver… Create prof… CSS based … Author: pluginlab.com	…oad	294,806	4/5	8-MX	Feb 01, 2007
PluginLab Horizontal Flyout Menu *** New Version 2 *** Create professional DHTML and CSS based Horizontal Menus Author: pluginlab.com	Download Win 907.4 KB	177,103	5/5	8-MX	Feb 01, 2007
Asp Upload You can upload files to Your site. This extension don&apos;t need any server components to upload files is a Pure Asp Upload. Author: gian61	Buy Win | Mac 167.2 KB	12,682	3/5	8-MX	May 19, 2006
Advanced Mail Advanced Mail lets you send e-mail from your web pages. It provides support for the most popular e-mail objects.	Buy Win | Mac 215.5 KB	19,770	3/5	8-MX	May 04, 2006

1-25 of 29

Figure 21-2: *You can peruse the Dreamweaver Exchange freely, check out the offerings, and even buy commercial third-party extensions. However, if you want to download one of the free extensions, you must get an Adobe ID and log into the site. Unfortunately, the marketing machine must be appeased, so you'll need to provide personal information and face a (fortunately optional) survey of your Web development habits.*

3. **Browse the extensions.**

   Once you've logged into the site, the home page highlights new and popular extensions. Overall, however, you may find the Exchange site a bit confusing and difficult to use. (For example, the entire site is in Flash, so Mac visitors can't use their browser's Back button to travel back to earlier screens.)

   Near the top of each page, you'll see a pop-up menu that lists different categories of extensions: Accessibility, DHTML/Layers, Navigation, Productivity, Flash media, Scripting, Tables, Text, eCommerce, and so on. When you choose a category, you go to a page listing all of the extensions within that category (see Figure 21-2).

   If you're looking for a *particular* extension, the Search command is your best bet. Click the Search Exchanges button (see Figure 21-2). On the Search page, type the extension's name or a few descriptive words into the Search Extensions field. Select Dreamweaver Exchange from the exchange menu (Adobe has exchanges for many products, including Fireworks, Flash, and Photoshop), and then click Search.

4. **Click an extension's name to go to its Web page.**

   On an extension's page, you'll find lots of information about the extension, such as a description of how it works, a button to either purchase or download the extension, information about which version of Dreamweaver and which operating system (Windows or Mac) it's compatible with, and buttons to add the extension to Favorites and Alerts lists. The Favorites option lets you create a personal list of the extensions you like best; the Alerts list feature means you'll receive an email whenever the extension is updated (see Figure 21-3).

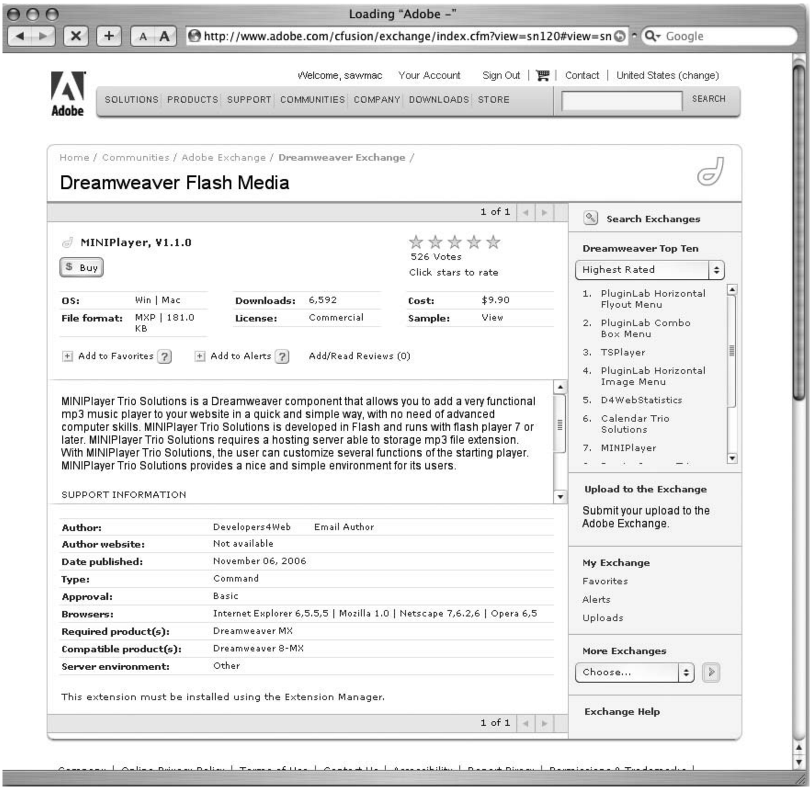
Find a Good Extension

How do you figure out which extensions are worth checking out? First, you can find some recommendations of the best ones scattered through this book in special boxes labeled Extension Alert (see page 166 and page 543, for instance).

The Exchange also provides information to help you separate the wheat from the chaff. Adobe tests each new extension before posting it. Extensions that pass a basic set of tests—it installs OK, it works, it doesn't blow up your computer—get a Basic approval rating. Some extensions also pass a more rigorous test that determines if the extension works in a way that's "Dreamweaver-like." In other words, these extensions are designed to look, feel, and act like the program, so that you won't need to learn a new interface. These extensions get an Adobe Approved rating, indicated by the word "Adobe" in the approval section of an extension's details page.

Of course, these approval ratings only let you know if an extension works; they don't tell you whether it's *useful*. As an extra aid, Dreamweaver aficionados (including you) can rate each extension on a scale of 1 (worst) to 5 (best) (see

Figure 21-3). An extension's average rating gives you a good indication of how handy it is. When you're browsing the Exchange, look for the column labeled with a star to the right (see Figure 21-2). Click the header to organize the extensions from most to least recommended.



OS:	Win | Mac	Downloads:	6,592	Cost:	$9.90
File format:	MXP | 181.0 KB	License:	Commercial	Sample:	View

MINIPlayer Trio Solutions is a Dreamweaver component that allows you to add a very functional mp3 music player to your website in a quick and simple way, with no need of advanced computer skills. MINIPlayer Trio Solutions is developed in Flash and runs with flash player 7 or later. MINIPlayer Trio Solutions requires a hosting server able to storage mp3 file extension. With MINIPlayer Trio Solutions, the user can customize several functions of the starting player. MINIPlayer Trio Solutions provides a nice and simple environment for its users.

Author:	Developers4Web Email Author
Author website:	Not available
Date published:	November 06, 2006
Type:	Command
Approval:	Basic
Browsers:	Internet Explorer 6,5.5,5 | Mozilla 1.0 | Netscape 7,6.2,6 | Opera 6,5
Required product(s):	Dreamweaver MX
Compatible product(s):	Dreamweaver 8-MX
Server environment:	Other

This extension must be installed using the Extension Manager.

Submit your upload to the Adobe Exchange.

Figure 21-3:
Each extension has its own page in Adobe Exchange that provides information about the extension and its developer. A helpful voting mechanism (1–5 stars) lets you know what other people think of the extension.

Other Extension Sources

Unfortunately, the glory days of totally free extensions are over. You can still find plenty of extensions offered free of charge, but many developers have realized they can't survive by giving away their work. The upside is that there are more excellent, polished, well-documented commercial extensions than ever—many even with customer support. Here are a few highlights:

- **WebAssist** (*www.webassist.com*) is one of the largest and most professional extension development companies. A former Dreamweaver product manager is at the helm, and they offer a wide variety of high quality extensions. (There are even a few free extensions to choose from.)
- If PHP or ASP (see page 751) is your bag, then you'll find an impressive collection of extensions at **Felix One** (*www.felixone.it/extensions/dwextensionsen.asp*).

- **Project Seven** (*www.projectseven.com*) offers free extensions and several excellent commercial extensions for creating animated Dynamic HTML and CSS menus, scrolling text areas, CSS-based page layouts, photo galleries, and more.
- **Trent Pastrana** (*www.fourlevel.com*) sells extensions for building photo galleries, whiz-bang DHTML effects (like menus that slide onto the page), or scrollers that move text up and down (or left and right) across a Web page.
- **Trio Solutions** (*http://components.developers4web.com/*) sells lots of inexpensive Dreamweaver extensions that let you use Dreamweaver to add CSS-style calendars, insert Flash music players into a page, and much more.

Download and Install Extensions

Once you've found a great extension, download it to your computer. You can save the downloaded file anywhere on your computer, but you might want to create a special folder. That way, if you ever need to reinstall Dreamweaver, you can quickly find and add your collection of extensions.

A downloaded extension file's name ends with *.mxp*, which stands for Macromedia Exchange Package (from the days before Adobe bought Macromedia). That's a special file format that works with the Extension Manager—the program, described next, that actually installs the extension into Dreamweaver.

Extension Manager

To add or remove a Dreamweaver extension, you use the Extension Manager, a standalone program that's integrated with Dreamweaver. It's designed to manage extensions for many Adobe programs (not just Dreamweaver): you can install extensions, turn them on and off, and remove them. This feature can be quite handy if you also use Adobe's Flash or Fireworks programs—you have a single access point for managing all your extensions.

You can launch the Extension Manager from within Dreamweaver by choosing Help → Manage Extensions, or Commands → Manage Extensions (see Figure 21-4).

To add an extension:

1. **Download an extension package (*.mxp* file) from the Exchange or another Web site.**

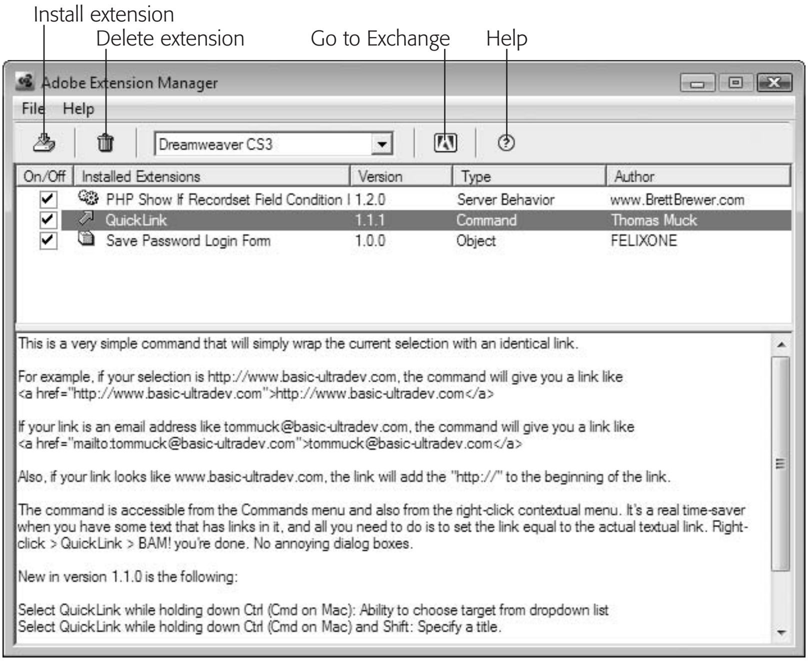
   See instructions on page 740.

2. **From Dreamweaver, choose Help → Manage Extensions (or Commands → Manage Extensions).**

   The Extension Manager launches. It lists all currently installed extensions (see Figure 21-4).

3. **Choose Dreamweaver CS3 from the pop-up menu.**

   Since the Extension Manager handles extensions for several different programs, you need to specify which program you're using. If you don't have any other Adobe products installed on your machine, Dreamweaver CS3 is the only option.



On/Off	Installed Extensions	Version	Type	Author
☑	PHP Show If Recordset Field Condition I	1.2.0	Server Behavior	www.BrettBrewer.com
☑	QuickLink	1.1.1	Command	Thomas Muck
☑	Save Password Login Form	1.0.0	Object	FELIXONE

This is a very simple command that will simply wrap the current selection with an identical link.

For example, if your selection is http://www.basic-ultradev.com, the command will give you a link like
<a href="http://www.basic-ultradev.com">http://www.basic-ultradev.com</a>

If your link is an email address like tommuck@basic-ultradev.com, the command will give you a link like
<a href="mailto:tommuck@basic-ultradev.com">tommuck@basic-ultradev.com</a>

Also, if your link looks like www.basic-ultradev.com, the link will add the "http://" to the beginning of the link.

The command is accessible from the Commands menu and also from the right-click contextual menu. It's a real time-saver when you have some text that has links in it, and all you need to do is to set the link equal to the actual textual link. Right-click > QuickLink > BAM! you're done. No annoying dialog boxes.

New in version 1.1.0 is the following:

Select QuickLink while holding down Ctrl (Cmd on Mac): Ability to choose target from dropdown list
Select QuickLink while holding down Ctrl (Cmd on Mac) and Shift: Specify a title.

Figure 21-4: *The Extension Manager window lists each extension you've installed, along with its version number, type, and author. If you select an extension from the list, a description appears in the bottom half of the window.*

4. **Choose File → Install Extension.**

   You can also click the Install Extension button. The Select Extension window appears, listing the folders on your hard drive.

5. **Navigate to and select the extension package (*.mxp* file) you wish to add.**

   A disclaimer appears with a lot of legal text. In brief, it frees Adobe from liability if your computer melts down as a result of installing the extension.

6. **Click Accept in the Disclaimer window.**

   A message may appear that asks you to quit and restart Dreamweaver. If so, follow the directions.

Tip: You can also install an extension simply by double-clicking the *.mxp* file after you download it, which saves you a few steps.

To remove an extension, select it from the list and choose File → Remove Extension, or click the Trash Can button.

Note: If you install a lot of extensions, Dreamweaver may take longer than usual to load; it needs to process every extension file as it opens. If you want to temporarily turn off an extension (as opposed to deleting it), open the Extension Manager and turn off the box next to the extension's name. To turn it back on, simply turn the box on again. You may need to restart Dreamweaver to make the extension available again.

Make Your Own Extensions

The Exchange is a great resource for finding useful extensions. But what if you can't find the extension you need? Create your own.

Writing extensions requires in-depth knowledge of JavaScript and HTML. But when you create a command that lets you complete a weekly task in a fraction of the time it previously took, the effort may just be worth it. For more information, choose Help → Extending Dreamweaver. The Adobe Web site also offers help at the Dreamweaver support center: *www.adobe.com/support/dreamweaver/extend.html*.

POWER USERS' CLINIC

The Secret Life of Extensions

Where do extensions go? The basic answer is: inside the Dreamweaver Configuration folder. But Dreamweaver actually supplies you with multiple configuration folders: the main folder located with the program itself, and account-specific folders for each user account on a computer. Windows and Macs let multiple users have an account on a single computer–one for you, your spouse, and your pet ferret, say. Of course, you may be the only one using your computer, so in that case there'd be just one additional configuration folder.

On a Windows machine, the main configuration folder is located in *C:\Program Files\Adobe\Adobe Dreamweaver CS3\configuration* (assuming the C:\ drive is your main drive). On a Mac, you can find it here: *Applications → Adobe Dreamweaver CS3 → configuration*. The individual account configuration folders are located in folders dedicated to each user. In Windows XP: *C:\Documents and Settings\[your user name]\Application Data\Adobe\Dreamweaver 9\Configuration*. In Windows Vista: *C:\Users\[your user name]\AppData\Roaming\Adobe\Dreamweaver 9\Configuration*. On a Mac: *Volume Name → Users → [your user name] → Library → Application Support → Adobe → Dreamweaver 9 → Configuration*. (Yes, the individual user folders refer to Dreamweaver 9, but it's really Dreamweaver CS3).

Some changes you make to Dreamweaver are recorded in your personal configuration folder, such as when you add an extension, delete a keyboard shortcut set (see page 733), or save a workspace layout (see page 27).

The main Configuration folder holds many of the files that control the look and operation of the program. For instance, the entire menu structure, including menu items and submenus, is described in a file called *menus.xml*. When Dreamweaver starts, it reads this file and uses the information inside it to draw the menus on the screen.

The Configuration folder holds many subfolders, each with a special purpose. For example, the Objects folder contains files that tell Dreamweaver which icon buttons appear on the Insert bar and how each one works.

Depending on the type of extension you've downloaded–command, object, behavior, or whatever–the Extension Manager stores the file (or files) required by the extension in one or more folders inside the Configuration folder. Because all of the files inside the Configuration folder are crucial to the way Dreamweaver works, don't delete the folder or any of the files inside it. In fact, because the Extension Manager automatically makes any required changes to the Configuration folder, there's no reason for you to even look inside it.

(The only exception is when you want to copy your keyboard shortcut set to another computer. See page 737.)

6

Part Six: Dynamic Dreamweaver

CHAPTER

22

Getting Started with Dynamic Web Sites

So far in this book, you've learned to build and maintain Web sites using Dreamweaver's powerful design, coding, and site management tools. The pages you've created are straightforward HTML, which you can immediately preview in a Web browser to see a finished design. These kinds of pages are often called *static*, since they don't change once you've finished creating them (unless you edit them later, of course). For many Web sites, especially ones where you carefully handcraft the design and content on a page-by-page basis, static Web pages are the way to go.

But imagine landing a contract to build an online catalog of 10,000 products. After the initial excitement disappears (along with your plans for that trip to Hawaii), you realize that even using Dreamweaver's Template tool (Chapter 19), building 10,000 pages is a lot of work!

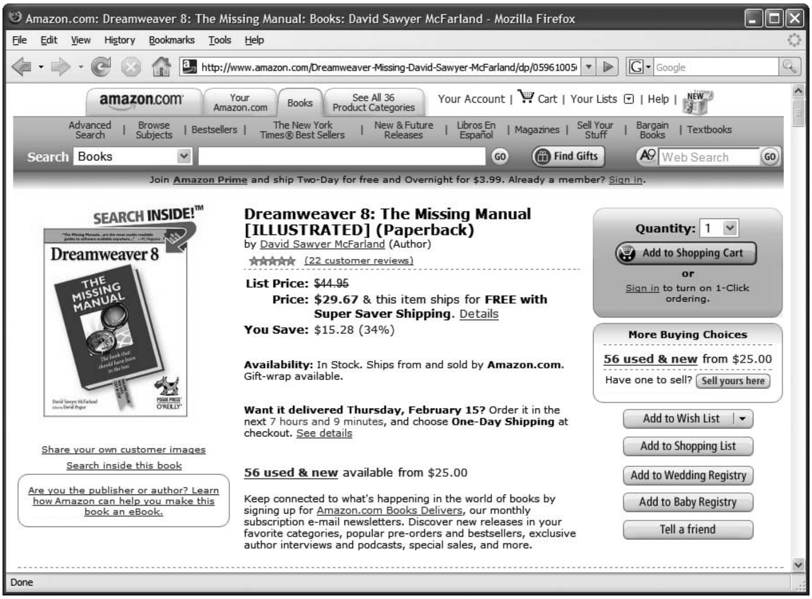
Fortunately, Dreamweaver offers a better and faster way to deal with this problem. Its dynamic Web site creation tools let you take advantage of a variety of powerful techniques that would be difficult or impossible with plain HTML pages. With Dreamweaver, you can build pages that:

- Display listings of products or other items like your record collection, your company's staff directory, or your mother's library of prized recipes.
- Search through a database of information and display the results.
- Require login so you can hide particular areas from prying eyes.
- Collect and store information from visitors to your site.
- Personalize your visitors' experience: "Hello Dave, it's been a while since you've visited. Did you miss us?—Hal."

Visit Amazon.com, for example, and you'll find more books than you could read in a lifetime. In fact, you'll find more products—DVDs, CDs, and even outdoor lawn furniture—than could fit inside a Wal-Mart. In just an hour, you could browse through hundreds of products, each with its own Web page. Do you really think Amazon hired an army of Web developers to create each Web page for every product they sell? Not a chance.

Note: Luckily you aren't limited to *either* "static" *or* "dynamic" Web pages. Web sites frequently contain both–static pages for custom designs and handcrafted content, and dynamic pages for mass production of a thousand catalog pages.

Instead, when you search for a book on Amazon.com, your search triggers a computer program, running on what's called an *application server*, which searches a large database of products. When the program finds products that match what you're searching for, it merges that information with the HTML elements that make up the page (banner, navigation buttons, copyright notice, and so on). You see a new Web page that's been created on the spot—perhaps for the first time ever (Figure 22-1).



Amazon.com: Dreamweaver 8: The Missing Manual: Books: David Sawyer McFarland - Mozilla Firefox

http://www.amazon.com/Dreamweaver-Missing-David-Sawyer-McFarland/dp/05961005

Join Amazon Prime and ship Two-Day for free and Overnight for $3.99. Already a member? Sign in.

Dreamweaver 8: The Missing Manual [ILLUSTRATED] (Paperback)
by David Sawyer McFarland (Author)

(22 customer reviews)

List Price: ~~$44.95~~
Price: **$29.67** & this item ships for **FREE with Super Saver Shipping.** Details
You Save: $15.28 (34%)

Availability: In Stock. Ships from and sold by **Amazon.com.** Gift-wrap available.

Want it delivered Thursday, February 15? Order it in the next 7 hours and 9 minutes, and choose **One-Day Shipping** at checkout. See details

56 used & new available from $25.00

Keep connected to what's happening in the world of books by signing up for Amazon.com Books Delivers, our monthly subscription e-mail newsletters. Discover new releases in your favorite categories, popular pre-orders and bestsellers, exclusive author interviews and podcasts, special sales, and more.

Quantity: 1

or

Sign in to turn on 1-Click ordering.

More Buying Choices

56 used & new from $25.00

Have one to sell?

Share your own customer images

Search inside this book

Are you the publisher or author? Learn how Amazon can help you make this book an eBook.

Done

Figure 22-1:
An infinite number of monkeys couldn't create all the Web pages for all the products Amazon sells. The solution? A dynamic Web site, which takes your programmed instructions (cooked up with a little help from Dreamweaver), and automatically creates pages made up of content chunks pulled from a database. That's the way to go if you've got a site with loads of pages that all present similar information.

Dynamic Web sites are usually the realm of professional programmers, but Dreamweaver can simplify routine tasks like viewing information from a database and adding, updating, and deleting data. Even if you don't have a programmer's bone in your body, this chapter and the next few give you the basics.

Pieces of the Puzzle

You may be thinking, "Yeah, that sounds fantastic, but so did that time-share in the Bahamas. What's the catch?"

The catch is that dynamic Web sites are more complex and require more technologies to get off the ground. Simple static Web sites require only the computer you use to build them, and a Web server to dish them out. In fact, as you can see by previewing your site from your own computer with a Web browser, you don't even need a Web server to effectively view a static Web site.

Dynamic Web pages, by contrast, require more (see Figure 22-2). Not only is there a Web server that handles requests for Web pages, two other types of servers enter the equation: an *application server* and a *database server*.

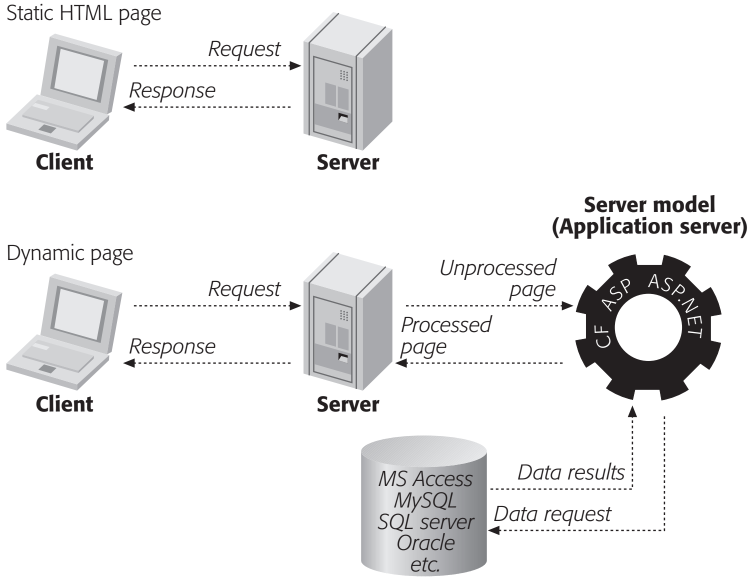


Figure 22-2: *Top: The technology behind a Web site that uses plain old static HTML pages is straightforward: When you come to the site, the Web server sends your computer (the client computer) the Web pages, graphics, and other elements of the page.*

Bottom: When you come to a dynamic page, on the other hand, the Web server must communicate with an application server, which in turn communicates with a database server. The final results come back to you through the Web server.

You'll still be using a lot of HTML (and CSS) to build a dynamic site—for example, to provide the layout, add banner graphics, and navigation bars. But you'll augment this mix with some form of programming code. The application server processes this code and sends a complete HTML page to the Web server, which, in turn, sends that onto the visitor. In many cases, the programming code requires the application server to retrieve information from a database, and then merge it with the HTML of a page.

Note: In this context, a *server* refers to software that dishes out particular types of information–Web pages, database queries, or a program's output. It doesn't necessarily mean a separate computer; Web hosting firms can (and frequently do) have Web, database, and application servers all running happily together on a single machine.

Because dynamic Web sites require more technology, you can't just open a dynamic page in your Web browser as you can a regular Web page. You must view a dynamic page through a Web server that has an appropriate application server running.

You also have to set up a database, and connect that database to your application server. Although this can be quite complex, it's not difficult to set up a basic Web server, application server, and database on your own computer, so that you can build and test database-driven Web pages. It's also easy to connect to other computers that are already configured to serve up dynamic, database-driven Web pages.

And once you or your company's system administrator have set up the Web server and other assorted components, Dreamweaver can easily create complex Web pages that access databases, and let you build powerful Web applications, all without ever learning any programming.

Note: The term *Web application* refers to Web pages that work together to complete a task. All the various pages that come together to form an online shopping site–which lets visitors do things like search a database of products, view individual product pages, and add products to a shopping cart–would be considered a Web application.

FREQUENTLY ASKED QUESTION

The Dynamic Duo

How does a dynamic Web site differ from dynamic HTML?

Dynamic is a word that's thrown around a lot in Web circles, and it has a variety of uses.

For starters, *dynamic* sometimes refers to the power of JavaScript. For example, Dreamweaver CS3's Spry Framework uses JavaScript to create interactive page elements such as the Spry menu bar discussed on page 175, or the animated effects described on page 511. The result is sometimes called "Dynamic HTML," because the elements on the page *change*.

However, in this section of the book, *dynamic* refers to any Web page that's processed by an application server–pages that undergo some form of transformation on the Web server's side of the Internet, like connecting to a database, or collecting information from a form.

What's important to remember is that JavaScript, used for Dynamic HTML, Spry, and Dreamweaver Behaviors (Chapter 13), is a *client-side* programming language. It runs in someone's Web browser, and is limited to changing the way a Web page looks and behaves *after* it's been downloaded over the Internet.

Dynamic Web sites, on the other hand, use *server-side* programs–those that run on an application server, out there on the Web somewhere. The dynamic part (responding to a form or accessing a database, for example) happens *on the Web server*. The visitors to your site never see any programming code, and their computers never have to run the program. They merely enjoy the results of the application server's hard work: a finished HTML page.

Even so, there are literally hundreds of combinations of Web, application, and database servers, and Dreamweaver doesn't work with all of them. However, it's capable of working with five of the most popular and powerful combinations, using seven different programming languages!

Understanding Server Models

In Dreamweaver lingo, the different application servers combine with a programming language to create a *server model*. Dreamweaver recognizes several server models, including ASP and .NET (pronounced "a-s-p" and "dot net"), ColdFusion, PHP, and JSP. Each server model has its own set of unique requirements, and its own methods of performing identical tasks.

Each server model also works with one or more programming languages. For example, you can create an ASP page using either VBScript or JScript. In some cases, the server model understands only a single programming language: JSP pages, for instance, use the Java programming language. And to make things just a bit more confusing, PHP can refer both to a programming language named PHP *and* to the application server. Likewise, CFML, or ColdFusion Markup Language, is a programming language, and ColdFusion server is an application server. If your head is hurting trying to make sense of all this, just keep this in mind: An application server processes programming code and carries out various actions, like talking to a database or spitting out a Web page.

Which server model you use depends on which resources you have available: which type of Web server hosts your site, the operating system it uses, and which application server is available. If you're hosting your site on Linux or Unix, you'll most likely end up using PHP; if you're hosting on Windows, meanwhile, you've already got access to ASP and probably .NET. It all comes down to what you have on your computer, what your company uses, or (if you're using a Web hosting service) what the host computer understands. Here's a brief description of each server model.

PHP

PHP (PHP Hypertext Preprocessor) is a programming language that was created specifically for building dynamic Web pages. It's the most popular and widely available option at Web hosting companies—in other words, when it comes time to place your finished Web site on the Internet, you're most likely to find a Web hosting company that supports PHP. (PHP is also quite often the least expensive hosting option.) The *PHP interpreter*—that's the application server—works in conjunction with a variety of Web servers, including Microsoft's Internet Information Server (IIS), but was initially created for the Apache Web server. PHP can also work with a variety of different database servers, but Dreamweaver understands only the MySQL database server (which is also available at nearly every Web hosting company).

Note: Because Apache, PHP and MySQL are so commonly used together, you may encounter the abbreviation frequently used to describe them–AMP.

Apache, PHP, and MySQL are free (one of the reasons they're so popular), and you can find simple installation programs that let you install all three programs on your own desktop computer. The tutorials for this section of the book use the PHP server model.

.NET

.NET is a Microsoft server technology. It's actually an entire suite of technologies intended to integrate many activities over the Internet.

.NET is also called ASP.NET because it was created as a replacement for Microsoft's older ASP (Active Server Page) technology. .NET can be programmed in numerous languages, including Microsoft's VB.NET, C# (pronounced "see sharp"), and JScript.NET, as well as more than 20 other languages. Dreamweaver recognizes only the C# and VB languages.

.NET runs in conjunction with Microsoft's Internet Information Server (IIS) Web server. IIS comes with Windows XP Professional and some versions of Windows Vista. Unfortunately, it doesn't come with Vista Home Basic, the most common version of Vista. You can find the .NET framework at *http://msdn2.microsoft.com/en-us/netframework/*.

.NET can work with a variety of different databases, with Microsoft's SQL Server being the most common.

ASP

ASP (Active Server Pages) used to be one of the most common ways to start building database-driven Web sites. It's a bit long in the tooth now, and is probably not the best choice if you're just starting with database-driven sites. ASP understands two different programming languages: VBScript and server-side JavaScript, both of which Dreamweaver speaks fluently. ASP also works with Microsoft's Web server—IIS.

ASP can work with a variety of databases. For small projects, you can use Microsoft Access (the database program that comes with some versions of Microsoft Office), since it's fairly easy to use. For more demanding projects, where you need to store lots of data, and many people will access your site, Microsoft's SQL Server is a better choice.

ColdFusion

ColdFusion is an application server from Adobe (the maker of Dreamweaver) that's programmed using CFML (ColdFusion Markup Language). ColdFusion works in conjunction with several different Web servers, including IIS and Apache, and uses its own programming language, which resembles HTML. For this reason, some Web designers find it easier to learn than other programming languages.

The downside is that this application isn't free. You *can* download a developer's edition—a free version that runs on your computer—so you can build and test ColdFusion Web pages. But if you want to host the Web site on the Internet, you have to either buy the ColdFusion Server package (which isn't cheap) or find a Web hosting company that offers ColdFusion hosting. Fortunately, in recent years, more and more Web hosting companies have started offering ColdFusion as an option, at rates that are close to or match regular hosting plans.

The Developer Edition of ColdFusion is available for download at *www.adobe.com/go/devcenter_cf_try*.

Like ASP and ASP.NET, ColdFusion works with many different databases.

JSP

JSP (JavaServer Pages) is based on Sun's popular Java programming language. It requires a Java application server, like Adobe's JRun server.

This approach isn't for the faint of heart, however, since setting up a Java Server and connecting it to a database can be tricky. Java is one of the more difficult languages to learn, too. If you don't have a knowledgeable guide, you're better off starting with a simpler technology like PHP.

A widely used version of JSP is Tomcat, available at *http://jakarta.apache.org/*. It's open source (that is, polished by a worldwide population of volunteer programmers), free, and it works with the Apache Web server and many different databases.

Picking a Server Model

With so many choices, you're probably wondering which server model to choose. If you've never built a dynamic Web site before, your best bet is PHP. You can easily set up a fully operational Web server (Apache), application server (PHP), and database server (MySQL) on your desktop computer, and quickly begin building dynamic pages with Dreamweaver.

In fact, since this is the easiest method, this book's tutorials concentrate on building PHP pages. Once you get the hang of Dreamweaver's dynamic Web-building tools, you can always try any of the other server models to build your sites. (However, switching a single site from one server model to another is difficult and not recommended.)

However, when you're building a real-world Web site, the final decision on which server model to use may be out of your hands. You may be working for a company that's already using ColdFusion for its Web site. Or, if you've already got a Web site up and running, but want to add some database-driven content, you have to use what's installed on that server. If your site is currently hosted at a Web hosting company, you should contact the company to find out which operating system, Web server, and databases it uses. If they're Windows-based, odds are that they use IIS, meaning that you can use ASP, or ASP.NET, and either Access or SQL Server databases. On the other hand, if they're a Unix operation, you'll most likely find the Apache Web server, PHP, and MySQL database.

Fortunately, the tools Dreamweaver provides for the different server models are largely the same. Essentially, you start to build dynamic Web pages using the same techniques you've learned in the earlier sections of this book. For the heavy lifting (like retrieving data from a database or password protecting a Web page), you'll turn to Dreamweaver's menu-driven database tools. They'll help you add the programming code necessary to make an application server do all the server-side magic you need to work with databases and generate dynamic Web pages. Once you learn Dreamweaver, you can build pages for any of the server models with which it works.

Dynamic Web Sites: The Setup

Now that your head is spinning, and you're considering some noble career alternative like farmer, firefighter, or carpenter, it's time to set up Dreamweaver to work with an application server and database.

You can do this setting up in several different ways. One involves using what Dreamweaver calls a *testing server*. Remember how you can create a Web site on your own computer (the *local site*) before posting it online for all to see (the *remote site*)? Here, the concept is similar. When building Web applications, it's again a good idea to keep all the "work in progress" pages on your own computer. After all, you don't want to fill up an active online database with test data, or put half-finished product pages on the Internet. But because dynamic Web sites require an application server and database, it's a good idea to create a *testing server* for storing and previewing dynamic pages in progress: a real Web server, application server, and database running on your own computer.

Then, when you've finished the site, you can transfer the pages to the remote site using Dreamweaver's built-in FTP feature (see page 610). If you're working in a group setting with other Web developers, you can set up the testing server on a machine that's part of your group's local network. Each developer can then connect to the testing server and retrieve files to work on. (Using Dreamweaver's Check-In/Check-Out feature [see page 628] is a good idea when you're working with a group of people on the same site.)

Note: You can always use your remote site as a testing server, as long as it has one of the application servers and databases that Dreamweaver works with. While this method is an easy way to get started, you must contact your Web host to see what application server it uses, and whether it can handle databases. In addition, you should have a fast Internet connection to the server. Otherwise, testing your dynamic pages may just test your patience.

Finally, whenever you work on dynamic files directly on a live Web server, be aware that mistakes you make along the way may affect a database that *other* dynamic pages use. If, while hurriedly trying to complete your Web site, you accidentally create a page that deletes records from your database, important information may no longer be available on your Web site. So whenever possible, the testing server should be separate from the server where the finished and perfected site resides.

In the next four chapters, you'll be building a dynamic Web site using PHP and a MySQL database. The concepts you learn work for all of the other server models as well, though some of the details may be different. Significant differences among various server models are mentioned where applicable.

To get started with the tutorials in this section of the book, you need to install Apache, PHP, MySQL, and a database-administration tool called phpMyAdmin. Don't worry, it's a lost easier than it may sound. There are several simple installers available for both Windows and Macs that make this step a snap. Because the procedure differs between Windows and Mac, each operating system has its own set of instructions below.

Setting Up a Testing Server for Windows

A quick way to install a testing server on Windows is to use software called *XAMPP*. XAMPP is free and works on both Windows XP and Windows Vista. It's basically a simple installer for putting Apache, PHP, and MySQL on your computer.

1. **Download the software at: *www.apachefriends.org/en/xampp-windows.html#641*.**

   Select the Installer option under the Basic Package. You may be taken to a page that presents you with a bunch of different download locations. Just click one of the download buttons, and then save the file to your desktop. Once downloaded, the installer works like most Windows installers.

   At the time of this writing, a direct link to the installer for the latest version of XAMPP (1.6.1) is located at: *www.apachefriends.org/download.php?xampp-win32-1.6.1-installer.exe*.

Warning: If you're already running Microsoft's IIS Web server on your computer, you must turn it off before continuing. XAMPP and IIS can't run together on the same computer without a fair amount of tinkering.

2. **Double-click the .exe file you downloaded.**

   A window opens, asking you to select the language you'd like to use.

3. **Choose a language from the menu, and then click OK.**

   A Setup Wizard window appears, ready to step you through the setup process.

4. **Click the Next button.**

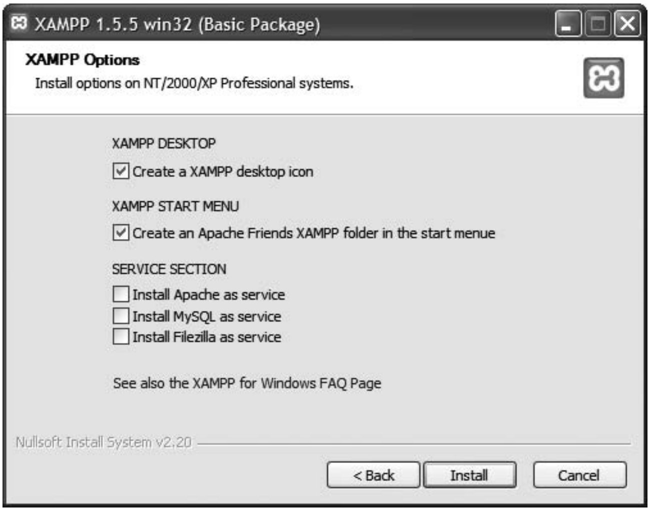
   The installer suggests putting the application on your main drive at C:\XAMPP. (The exact location the installer recommends may vary: Previous versions of the installer recommended C:\Program Files\XAMPP.) You can pretty much install it anywhere, but putting it in the Program Files folder on your main drive (usually C:\Program Files) where other programs are installed is usually a good idea.

5. **Click the Next button once again.**

   The XAMPP Options window appears (Figure 22-3). In most cases, it's fine to leave all the window's checkboxes just as you see; see Figure 22-3 for details.

6. **Click Install.**

   The installer places all the files onto your system. This process takes a while, since a lot of programs and files are being installed.



XAMPP 1.5.5 win32 (Basic Package)

XAMPP Options

Install options on NT/2000/XP Professional systems.

XAMPP DESKTOP

☑ Create a XAMPP desktop icon

XAMPP START MENU

☑ Create an Apache Friends XAMPP folder in the start menue

SERVICE SECTION

□ Install Apache as service
□ Install MySQL as service
□ Install Filezilla as service

See also the XAMPP for Windows FAQ Page

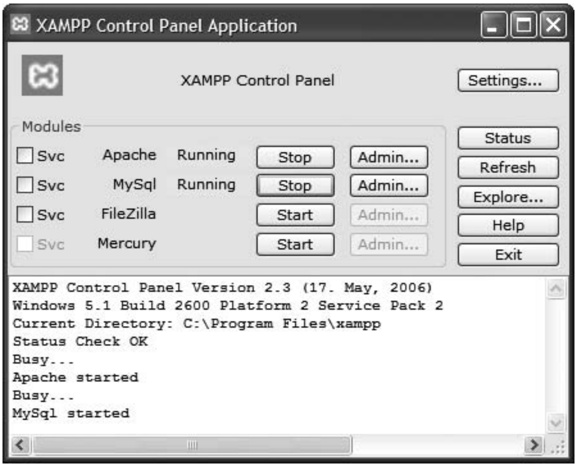
Figure 22-3:
If you plan on doing a lot of development, day in and day out, you might want to turn on the "Install Apache as service" and "Install MySQL as service" checkboxes. A service starts up every time you turn on your computer, so Apache, PHP, and MySQL are always running. However, if you won't be building database sites frequently, or you don't have a lot of RAM in your computer, don't turn on these boxes (you'll just have to manually start the servers when you wish to build dynamic pages, using the XAMPP control panel described on the next page).

7. **Finally, click the Finish button.**

   A window appears "congratulating" you (way to double-click the installer program!), and asking whether you wish to start the XAMPP Control panel.

8. **Click Yes, to open the XAMPP Control Panel** (**see Figure 22-4**).

   The XAMPP Control Panel lets you start and stop the Apache Web server and MySQL database server.



XAMPP Control Panel Application

XAMPP Control Panel

Modules

□ Svc Apache Running
□ Svc MySql Running
□ Svc FileZilla
□ Svc Mercury

```
XAMPP Control Panel Version 2.3 (17. May, 2006)
Windows 5.1 Build 2600 Platform 2 Service Pack 2
Current Directory: C:\Program Files\xampp
Status Check OK
Busy...
Apache started
Busy...
MySql started
```

Figure 22-4:
In this figure, both Apache and MySQL are currently running, as indicated by the word Running to the right of their names. Click the Stop buttons to turn the servers off. You can open the Control Panel by clicking the XAMPP Control Panel shortcut on your desktop.

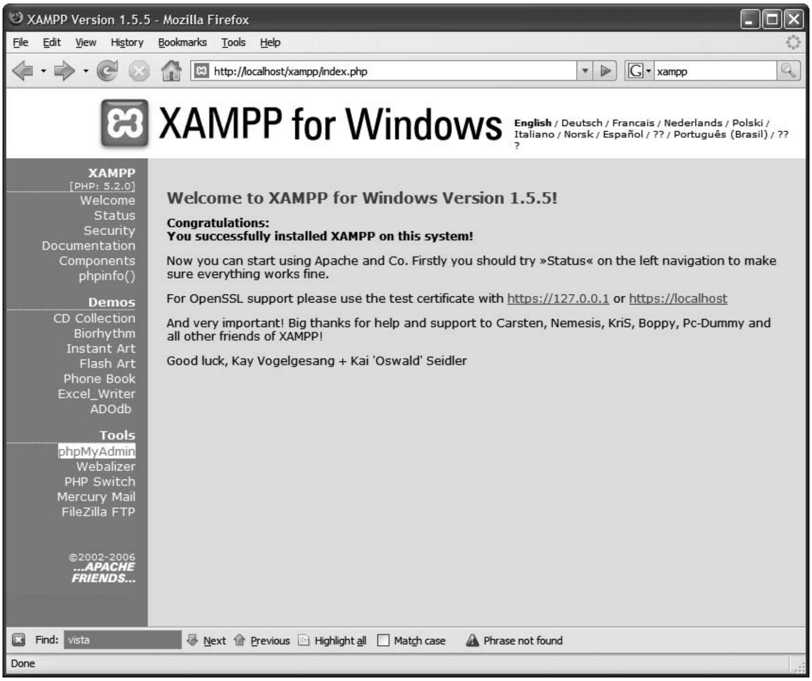
9. **If the buttons to the right of Apache and MySQL say Start, click them to start the Web server and the MySQL database server.**

You probably get a Windows security alert about both MySQL and Apache: Click the Unblock button in both cases. This action allows the two servers to run, and tells the Windows firewall protection service that everything is OK.

If Apache and MySQL are already running, these buttons say Stop. (Clicking them turns off the Web server and MySQL.) Whenever you start Apache, PHP automatically starts as well. At this point, you should have a complete testing server running on your machine. You just need to make sure it's working.

10. **To do so, launch a Web browser, and, in the Location bar, type *http://localhost/*.**

You encounter a page that lists a bunch of languages; click the language you prefer, and you're taken to a kind of Web-based control panel for XAMPP (Figure 22-5).



Welcome to XAMPP for Windows Version 1.5.5!

Congratulations:
You successfully installed XAMPP on this system!

Now you can start using Apache and Co. Firstly you should try »Status« on the left navigation to make sure everything works fine.

For OpenSSL support please use the test certificate with https://127.0.0.1 or https://localhost

And very important! Big thanks for help and support to Carsten, Nemesis, KriS, Boppy, Pc-Dummy and all other friends of XAMPP!

Good luck, Kay Vogelgesang + Kai 'Oswald' Seidler

Figure 22-5: *Once installed, you can view your XAMPP home page from http://localhost/xampp/. From the left-hand list of links, you can access helpful programs and information, such as phpMyAdmin (for working with the MySQL database) and phpinfo() for finding out more about the server setup.*

Once you've installed XAMPP, you'll see a shortcut called XAMPP Control Panel on your desktop. Double-click this icon to control the servers you've just installed—you can turn the servers off and on, as well as turn them into services (which launch each time you start up your computer).

Note: To uninstall XAMPP, just go to the location where you installed XAMPP (like C:\XAMPP\ or C:\Program Files\XAMPP\) and run the program named Uninstall.exe. This action, however, deletes any databases you created, and destroys any Web pages that you placed on the server. To prevent this process, just follow these steps: First, use phpMyAdmin to export any database you wish to save (you can find instructions at *http://php.about.com/od/learnmysql/ss/mysql_backup_3.htm*), and then make a copy of the files located in the C:\Program Files\XAMPP\htdocs folder.

For more information on XAMPP, including even more detailed installation and setup instructions, visit *www.apachefriends.org/en/xampp-windows.html*.

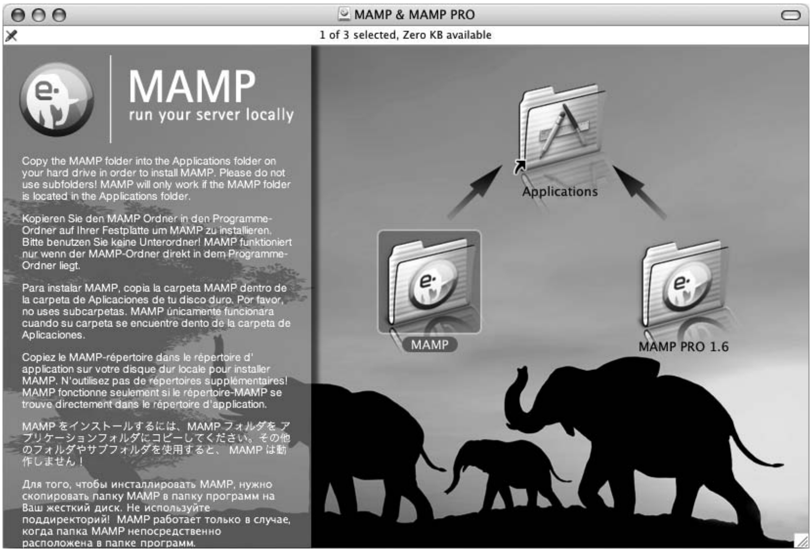
Setting Up a Testing Server for Mac OS X

Although XAMPP comes in a Mac OS X version, it's not really the best choice for Mac people. XAMPP for Mac isn't as well updated as its Windows counterpart, and the initial setup isn't very Mac-like. MAMP, which is also free and easy to install, is a better option.

1. **Download the software at: *www.living-e.com/products/MAMP-PRO/download.php*.**

   Under the heading Installation Packages, you should see a link for MAMP and MAMP Pro 1.6 (or whatever the current version is). It's a big download, so it takes a while to complete. (MAMP is free, but the download also includes the commercial product, MAMP Pro, which makes administering MAMP easier. You don't need MAMP Pro for this tutorial, and you don't have to install it either.)

   Once downloaded, the file (named something like MAMP_1.6_universal.dmg.zip) unzips itself, and appears in the list of volumes in a Finder window. In addition, the window pictured in Figure 22-6 should appear. (You may need to double-click the .zip file to unzip its contents.)



Copy the MAMP folder into the Applications folder on your hard drive in order to install MAMP. Please do not use subfolders! MAMP will only work if the MAMP folder is located in the Applications folder.

Kopieren Sie den MAMP Ordner in den Programme-Ordner auf Ihrer Festplatte um MAMP zu installieren. Bitte benutzen Sie keine Unterordner! MAMP funktioniert nur wenn der MAMP-Ordner direkt in dem Programme-Ordner liegt.

Para instalar MAMP, copia la carpeta MAMP dentro de la carpeta de Aplicaciones de tu disco duro. Por favor, no uses subcarpetas. MAMP únicamente funcionara cuando su carpeta se encuentre dento de la carpeta de Aplicaciones.

Copiez le MAMP-répertoire dans le répertoire d' application sur votre disque dur locale pour installer MAMP. N'outilisez pas de répertoires supplémentaires! MAMP fonctionne seulement si le répertoire-MAMP se trouve directement dans le répertoire d'application.

MAMP をインストールするには、MAMP フォルダを アプリケーションフォルダにコピーしてください。その他のフォルダやサブフォルダを使用すると、 MAMP は動作しません！

Для того, чтобы инсталлировать MAMP, нужно скопировать папку MAMP в папку программ на Ваш жесткий диск. Не используйте поддиректорий! MAMP работает только в случае, когда папка MAMP непосредственно расположена в папке программ.

***Figure 22-6:** Installing MAMP is just a matter of downloading a DMG (disk image) file, unzipping it, and dragging the MAMP folder into your Applications folder.*

2. **Drag the MAMP folder (not the MAMP Pro folder) into your Applications folder (see Figure 22-6).**

   That's it! You've just installed the world's most popular and powerful Web server (tell your IT guys that the next time they say it'll take two weeks for them to fix your email problem).

Now, you need to start the Web server.

3. **In Applications → MAMP, start the MAMP program, and then click the Start Servers button if a red light appears to the left of either Apache or MySQL (see Figure 22-7).**

   The MAMP program (see Figure 22-7) lets you start and stop the Web and MySQL servers, as well as set preferences for how each server works.

   At this point, you have a functioning Web server, database server, and application server on your Mac. (PHP starts automatically whenever you start Apache.) You need to take care of one last thing before seeing if the server works; step 4 has the details.

Tip: Since you'll frequently access the MAMP program to turn the Web server on and off, it's a good idea to add it to the OS X Dock.

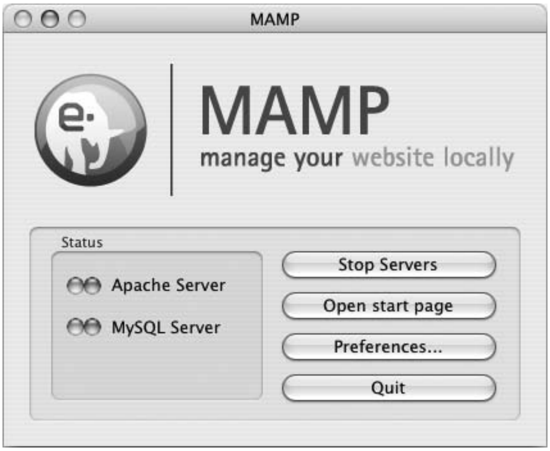
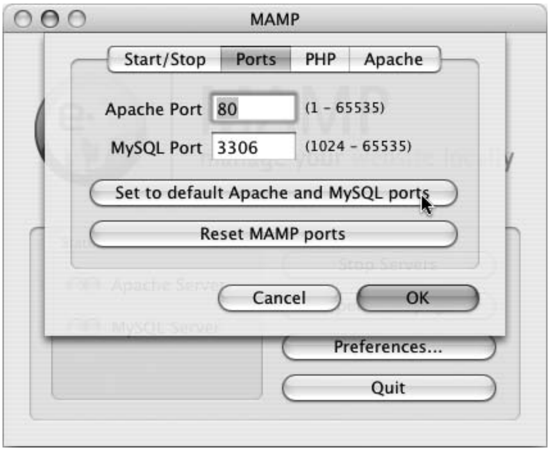


Figure 22-7:
When the servers are running, a green button to the left of Apache Server and MySQL Server lights up. To stop the servers, click the Stop Servers button. A red light next to either server indicates that the server is turned off.

4. **In the MAMP program, click the Preferences button, and then click the Ports tab (see Figure 22-8).**

   Ports are virtual entryways that manage how other computers access different server programs on your Mac. For example, you can assign a Web server to a particular port number, and then all requests for Web pages are directed to that port. By convention, port 80 is reserved for Web servers; Web browsers therefore normally request Web pages over that port.

   When you visit *www.cosmofarmer.com*, you're really visiting port 80 of the server that's hosting *www.cosmofarmer.com*. Port 3306 is most commonly used for MySQL. However, out of the box, MAMP uses ports 8888 and 8889 for Apache and MySQL, respectively. Unfortunately, this system means you need to tack on the port number whenever you request a Web page from your testing server. So instead of typing *http://localhost/* into a Web browser, to go to the home page for your local server, you must type *http://localhost:8888/*. This is a bit of a pain. Fortunately, it's easy to change.



Apache Port 80 (1 - 65535)
MySQL Port 3306 (1024 - 65535)

Figure 22-8:
After installing MAMP it's a good idea to set Apache and MySQL to their normal port settings of 80 and 3306. You can't do this setting, however, if you don't have administrator privileges for your computer. In that case, you have to stick to the 8888 and 8889 port settings that MAMP ships with. When you visit a Web page running on your local server, you have to add the port, like this: http://localhost:8888/my_cool_page.php.

5. **Click the "Set to default Apache and MySQL ports" button (see Figure 22-8), and then click OK.**

   You need administrator privileges to make these port changes, so you're prompted to enter your administrator's password. (If you're not using an administrator's account, see Figure 22-8.) Now it's time to see if everything worked.

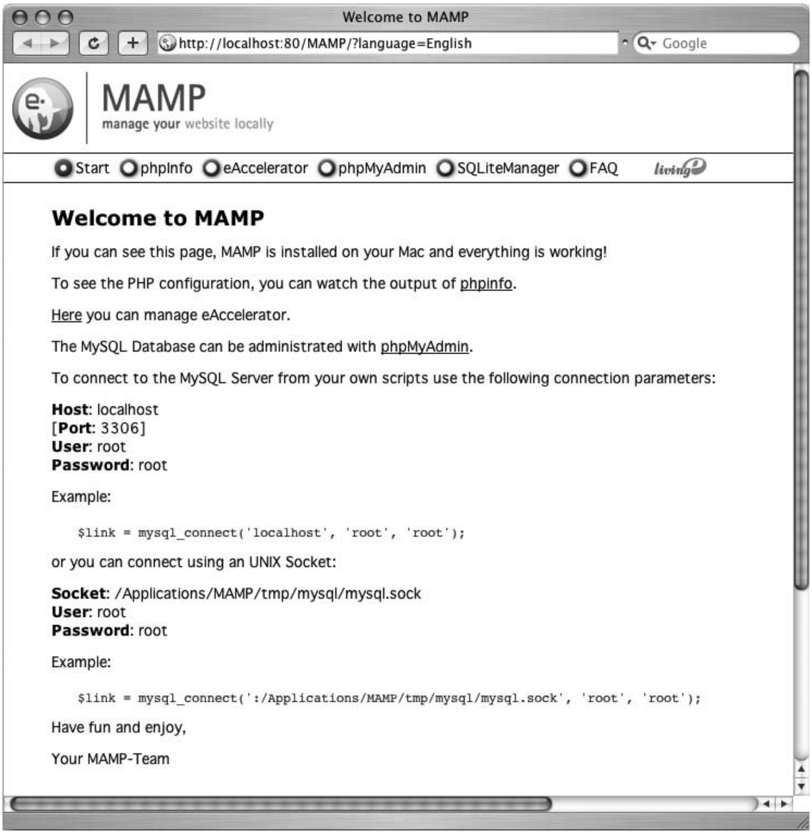
Note: Mac OS X ships with a version of Apache already installed. If you have it running when you install MAMP, you can't set the MAMP Apache port to 80. In this case, turn off Apache on your Mac by opening System Preferences. Click the Sharing tab, and then turn off the Personal Web Sharing checkbox. Now you can complete steps 4 and 5 above to set up MAMP's Apache on port 80.

6. **In the MAMP program window, click the "Open start page" button (see Figure 22-7).**

   The home page for your new MAMP installation appears (see Figure 22-9). If you see the page, you know everything is set up correctly—you're actually viewing this page through an active Web server on your own computer (pretty cool and geeky stuff). From this page you can access phpMyAdmin, a Web-based tool for managing a MySQL database (you'll be using phpMyAdmin to help set up the database for the tutorials).

   You can find a helpful forum for asking questions and learning more about MAMP at *http://forum.mamp.info/index.php?c=2*.

To remove MAMP, simply drag the MAMP folder into the trash; but first remember to remove any Web page files you've created in the *htdocs* folder, and back up your MySQL database if you want to keep the data in it. You can learn how to back up a MySQL database at *http://php.about.com/od/learnmysql/ss/mysql_backup_3.htm*.



Welcome to MAMP

If you can see this page, MAMP is installed on your Mac and everything is working!

To see the PHP configuration, you can watch the output of phpinfo.

Here you can manage eAccelerator.

The MySQL Database can be administrated with phpMyAdmin.

To connect to the MySQL Server from your own scripts use the following connection parameters:

Host: localhost
[**Port**: 3306]
User: root
Password: root

Example:

```
$link = mysql_connect('localhost', 'root', 'root');
```

or you can connect using an UNIX Socket:

Socket: /Applications/MAMP/tmp/mysql/mysql.sock
User: root
Password: root

Example:

```
$link = mysql_connect(':/Applications/MAMP/tmp/mysql/mysql.sock', 'root', 'root');
```

Have fun and enjoy,

Your MAMP-Team

Figure 22-9:
The MAMP home page gives you access to a few useful tools, including phpMyAdmin, which you'll use for working with MySQL databases. You can always get to the MAMP homepage by typing its URL in a Web browser: http://localhost/MAMP; or, in the MAMP controller program, clicking the Open Start Page button (Figure 22-7).

Localhost and the Htdocs Folder

If you followed the previous instructions and installed a testing server on your computer, you've already visited a Web page at either *http://localhost/xampp* or *http://localhost/MAMP* (the home pages for XAMPP or MAMP). You may be wondering, what's this *localhost* thing? For a computer, "localhost" is just another way of saying "me." When you instruct a browser to go to *http://localhost*, you're merely telling it to look for a Web server running on the same computer as the Web browser. Normally, when you visit a Web site, you type a Web address like *http://www.google.com/*. That sends your Web browser out over the Internet looking for a Web page located on some computer identified as *www.google.com*. When you've set up a Web server on your own computer, and you wish to view the Web pages you've placed there, the Web browser need look no further than your own computer.

But once the browser asks the local Web server to give it a Web page, where does the Web server find that page on your computer? When working with static Web pages (like the ones you built earlier in this book), you can keep your Web site files pretty much anywhere you want: on your desktop, in your Documents folder, on an external hard drive, and so on. Dynamic pages, on the other hand, work only with a Web server, and must reside in a particular location on your computer in order for the Web server to find them.

That folder is called the *site root* folder (you may also hear it referred to as the *document root)*. The exact name and location of the site root folder varies from system to system. For example, different Web hosting companies have different setups, and might name the folder *htdocs*, *webdocs*, or *public_html*. XAMPP and MAMP both use a folder named *htdocs* as the site root folder. For XAMPP users, that folder is located in the *htdocs* folder in your XAMPP installation (C:\Program Files\xampp\htdocs, for example); for MAMP folks, head over to Applications → MAMP → htdocs.

In the case of XAMPP, if you type *http://localhost/my_page.html* into your browser, the browser requests a file named *my_page.html* from the Web server running on your computer. The Web server then looks inside C:\Program Files\xampp\htdocs for a file named *my_page.html*; if it finds it, the server sends the file back to the browser. On a Mac running MAMP, the Web server would look in Applications → MAMP → htdocs for the file *my_page.html*.

Note: If you don't specify a particular file—for example, you just surf to *http://localhost/*—the Web server looks for a "default file" (usually named *index.html* or *index.php*) as described on page 560.

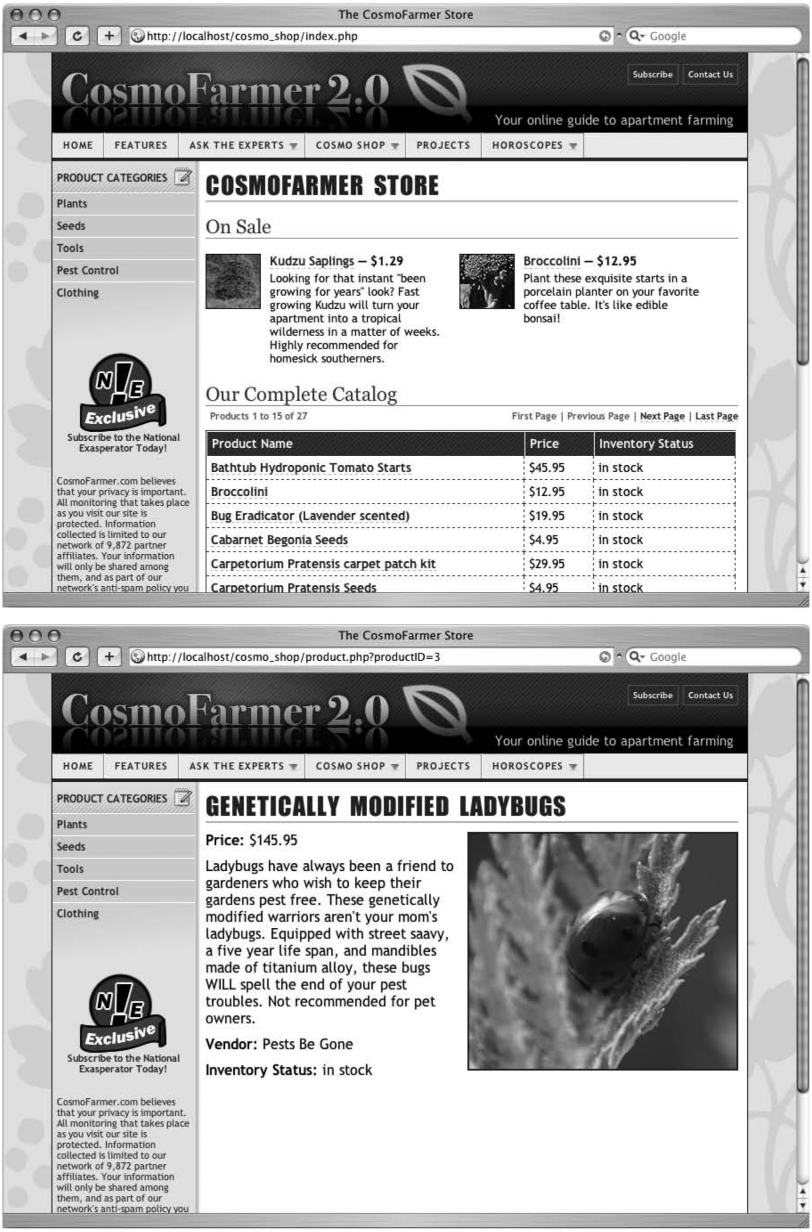
Remember, when you're working on a dynamic, database driven site, you need to keep your Web site files inside the site root folder for your testing server. If you don't, Dreamweaver doesn't let you start building database-driven pages.

Tip: You can also put your site files in a folder *inside* the site root folder. If you placed a folder named *store* in the *htdocs* folder, you could visit a Web page named *products.php* inside that folder by browsing to *http://localhost/store/products.php*.

Setting Up Dreamweaver

To learn how to use Dreamweaver's dynamic features, you'll be building a small Web application for CosmoFarmer.com (see Figure 22-10). In fact, you'll turn the site's online store into a group of dynamic Web pages that retrieve information from a database, and merge it with existing HTML code.

Before you begin building the page, download the tutorial files. As always, you can find them at *www.sawmac.com/dwcs3/*; click the Tutorials link to go to the tutorials page and download the files. If you've done any of the previous tutorials, you've already downloaded the necessary files. The files for this section of the book are inside the *php_dynamic* folder, which is located inside the *MM_DWCS3* folder. Inside the *php_dynamic* folder you'll find another folder named *cosmo_store*, and a file named *cosmofarmer.sql*.



The CosmoFarmer Store

http://localhost/cosmo_shop/index.php

Subscribe to the National Exasperator Today!

CosmoFarmer.com believes that your privacy is important. All monitoring that takes place as you visit our site is protected. Information collected is limited to our network of 9,872 partner affiliates. Your information will only be shared among them, and as part of our network's anti-spam policy you

COSMOFARMER STORE

On Sale

Kudzu Saplings — $1.29
Looking for that instant "been growing for years" look? Fast growing Kudzu will turn your apartment into a tropical wilderness in a matter of weeks. Highly recommended for homesick southerners.

Broccolini — $12.95
Plant these exquisite starts in a porcelain planter on your favorite coffee table. It's like edible bonsai!

Our Complete Catalog

Products 1 to 15 of 27 First Page | Previous Page | Next Page | Last Page

Product Name	Price	Inventory Status
Bathtub Hydroponic Tomato Starts	$45.95	in stock
Broccolini	$12.95	in stock
Bug Eradicator (Lavender scented)	$19.95	in stock
Cabarnet Begonia Seeds	$4.95	in stock
Carpetorium Pratensis carpet patch kit	$29.95	in stock
Carpetorium Pratensis Seeds	$4.95	in stock

The CosmoFarmer Store

http://localhost/cosmo_shop/product.php?productID=3

Subscribe to the National Exasperator Today!

CosmoFarmer.com believes that your privacy is important. All monitoring that takes place as you visit our site is protected. Information collected is limited to our network of 9,872 partner affiliates. Your information will only be shared among them, and as part of our network's anti-spam policy you

GENETICALLY MODIFIED LADYBUGS

Price: $145.95

Ladybugs have always been a friend to gardeners who wish to keep their gardens pest free. These genetically modified warriors aren't your mom's ladybugs. Equipped with street saavy, a five year life span, and mandibles made of titanium alloy, these bugs WILL spell the end of your pest troubles. Not recommended for pet owners.

Vendor: Pests Be Gone

Inventory Status: in stock

Figure 22-10: *Whenever you need to display lots of similar information, dynamic Web pages may be the answer. Dynamic pages at the CosmoFarmer online store list many different products. Because all the product information is stored in a database, it takes only two dynamic pages to display a complete list of products (top), as well as detailed information for each individual product (bottom).*

To begin, move the *cosmo_store* folder into the newly installed Web server's root folder. If you followed the previous directions, the root folder on Windows should be in C:\Program Files\xampp\htdocs, while the root folder for Macs is in Applications → MAMP → htdocs. Place *cosmo_shop* inside the *htdocs* folder. To make sure you've set this up right, open a Web browser, and then, in the Address bar, type *http://localhost/cosmo_shop/*. If a Web page appears, your Web server is set up correctly.

The first step in working on this dynamic Web application is defining a new site. The process of defining a dynamic site, as outlined in the following steps, is slightly different than for static sites, but no harder:

1. **Start Dreamweaver, and then choose Site → New Site.**

   The Site Definition window opens. Because there are more things to keep track of when setting up a dynamic site, it's easiest to use Dreamweaver's Site Wizard.

2. **If it isn't already selected, click the Basic tab at the top of the window.**

   First, you need to give this new site a name.

3. **Type *Cosmo Shop* in the first box, and *http://www.cosmofarmer.com/cosmo_shop/* in the second box.**

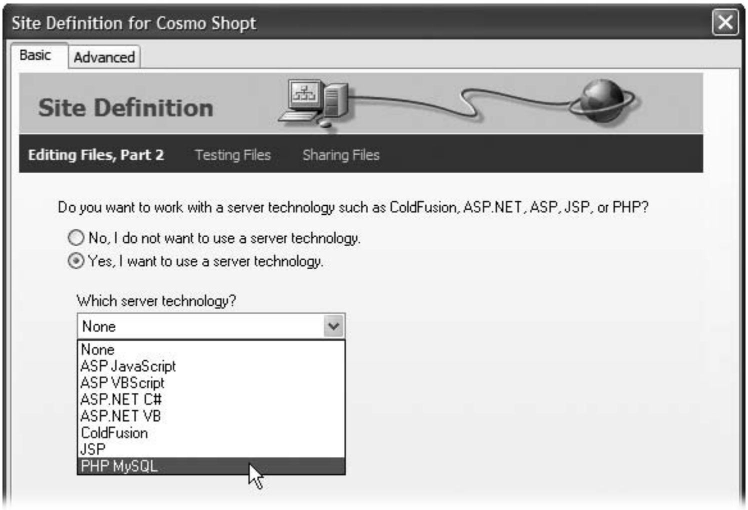
   You've just told Dreamweaver the name you want to use while working on this site, and the URL of the Web site. In this case, you're actually just working on one part of the CosmoFarmer site, the store, which is located in a folder (*cosmo_shop*) within the main site. In a real-world scenario, you'd type the address of your Web site.

4. **Click Next.**

   The next screen lets you choose whether you're building a static or dynamic Web site.

5. **Select "Yes, I want to use a server technology," and then, from the pop-up menu, choose "PHP MySQL" (see Figure 22-11). Click Next to proceed.**

   In the next step, you'll tell Dreamweaver where your local files are, and where you intend to put the files for the testing server.



Site Definition for Cosmo Shopt

Do you want to work with a server technology such as ColdFusion, ASP.NET, ASP, JSP, or PHP?

○ No, I do not want to use a server technology.

● Yes, I want to use a server technology.

Which server technology?

None

Figure 22-11: *When building a dynamic site, you must choose one of seven different server models. There are actually only five different application servers, but since Dreamweaver lets you use two different languages for both ASP and ASP.NET, there are two server models for each of those application servers.*

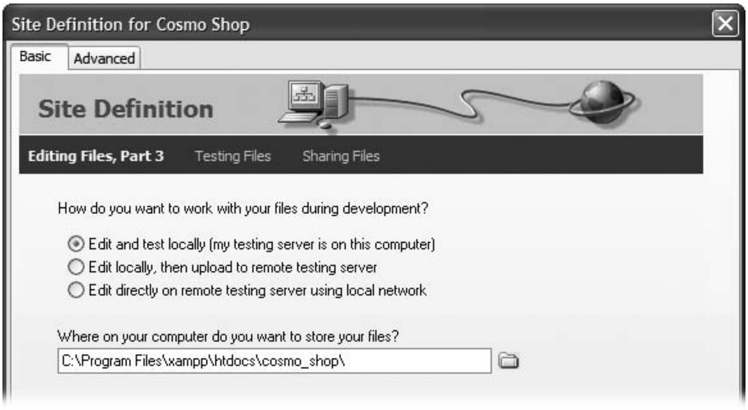
6. **Select "Edit and test locally"** (see Figure 22-12).

Dreamweaver provides three ways to work with dynamic Web page files and a testing server.

"Edit and test locally" is a good choice when you've set up a Web and application server on your computer (as you've done in this tutorial). Essentially, you're working on Web pages located on a functioning Web server. In this way, you preview the pages running on a real Web server, so you can immediately test out all the nifty dynamic stuff.

Use the other two options when the testing server is located on another computer. This computer may be one on your local network or a full-fledged Web server running on the Internet that you connect to using FTP.

"Edit locally, then upload to remote testing server" is a good option if you already set up a Web site with a Web hosting company, and you can't run a testing server on your computer—for example, if you're building ASP pages but you're on a Mac.



How do you want to work with your files during development?

● Edit and test locally (my testing server is on this computer)
○ Edit locally, then upload to remote testing server
○ Edit directly on remote testing server using local network

Where on your computer do you want to store your files?

C:\Program Files\xampp\htdocs\cosmo_shop\

Figure 22-12: *Dreamweaver provides different ways to work with dynamic Web page files on a testing server. If you don't have a working Web server on your computer, but do have an account with a Web hosting company that recognizes one of Dreamweaver's server models, you can use that server to test your files. Choose the second option: "Edit locally, then upload to remote testing server." This method is a bit laborious–you edit the pages on your computer; then, when you test them (a simple F12 [control-F12] to see if they'll work), Dreamweaver connects to your Web server, uploads the file, launches your Web browser, and loads the page from your Web site. But it's one way to get started building dynamic Web sites.*

People don't use the last option often, but if your company happens to have its own in-house Web server, to which you can connect over a local network, then you can use this option. But, in general, it's best not to work on a live Web server—your works-in-progress might be viewable by anyone with a Web browser. In addition, if more than one person is working on the Web site at the same time, you should use some kind of version control system to make sure no two people are working on the same file at the same time (Dreamweaver's Check In/Check Out system is one method—see page 628).

The next step involves telling Dreamweaver where to find the files for the Web site.

7. **Click the folder at the right side of the middle of the window; navigate to and select the cosmo_shop folder on the Web server. Click Next.**

   If you're following along using XAMPP, you'll find the *cosmo_shop* folder at C:\Program Files\xampp\cosmo_shop; MAMP folks can find it at Applications → MAMP → htdocs → cosmo_shop.

8. **Type *http://localhost/cosmo_shop/* in the box, and then click Test.**

   Dreamweaver may have already filled in this box. If the test server is running on your computer, the URL begins with *http://localhost/* and ends with the folder that contains the Web pages. In this case, the URL is *http://localhost/cosmo_shop/*.

Note: If you're running MAMP and can't change the port Apache uses as described on page 760, you need to add the port number 8888 to the URL, like this: *http://localhost:8888/cosmo_shop/*.

   If you get an error message when you click Test, you've probably entered the wrong URL. You can make this mistake when the folder following *localhost* in the URL is not actually in the site root folder of the testing server, or the Web server isn't running (to start the XAMPP Web server, see page 757; MAMP people turn to page 759).

9. **Click Next. Click No, and then click Next one more time.**

   If you were planning to move this site onto a Web server connected to the Internet, you would select Yes at this stage, and provide all the information needed to move your site files to the Internet as described in Chapter 17. But since this tutorial is just an exercise, you won't be putting it up on a live Web server.

10. **Click Done.**

    Dreamweaver has successfully set up your site. You're now ready to learn about databases, and set up a new database using the MySQL server.

Creating a Dynamic Page

Once you've set up an application server and a database server, you're ready to connect to a database, retrieve information, and display it on a Web page.

You already know how to handle the first step: Design an HTML page to display the database information. Dynamic pages differ from regular HTML pages in a couple ways. For starters, the name of a dynamic file doesn't end with .html. Depending on which server model you use, dynamic pages end in .php (for PHP pages), .asp (ASP), .aspx (.Net), .cfm or cfml (Cold Fusion), or .jsp (JavaServer

Pages.) The file extension you use is important: A Web server uses it to identify the type of page requested. If a Web server gets a request for an .html file, it simply finds it and sends it to the Web browser. But if it gets a request for a page that ends in, say, .php, it will send the page to the application server to sort out all the messy programming.

The good news is that the basic process of creating a new, blank, dynamic page is the same as with a regular HTML page:

1. **Choose File → New to open the New Document window. Select the Blank Page category; from the Page Type list, choose a dynamic page type (PHP, ASP VBScript, ColdFusion, or whatever).**

2. **From the Layout list, choose a layout (or none if you wish to start with a fresh, blank page), and then click the Create button.**

   When you save the file, Dreamweaver automatically adds the proper extension: .asp for ASP pages, .aspx for ASP.NET, .cfm for ColdFusion, .jsp for JSP, or .php for PHP pages.

3. **Right-click (Control-click) in the Site panel; choose New File from the shortcut menu.**

   Dreamweaver creates a file in the correct server model format, with the proper extension.

Note: Just renaming a file in the Sites panel (from *about.html* to *about.asp*, for example) does *not* give the file the code necessary to apply the correct server model to the page. This fact holds true for ASP, .NET, and JSP pages. (However, PHP and ColdFusion pages don't start life within any special code, so you could start with a .html page, change the extension to .php, and then add PHP programming.) More importantly, changing the file's extension (from .asp to .php, for example) doesn't change the page to the new server model, either, and usually ends up "breaking" the page.

Once you've created the page, you can then use any of the page-building tools described in this book—Cascading Style Sheets, Spry Widgets, Library items, or whatever—to design the page. Even though the file's officially a PHP page, it still contains lots of HTML. Unlike a plain-vanilla HTML page, though, this one can also contain the server-side programming code that lets the page communicate with a database.

Finally, you can also edit the newly created page using either Design view or Code view. But before you can add dynamic content to a page, you need to create a connection to a database.

Databases: A Quick Introduction

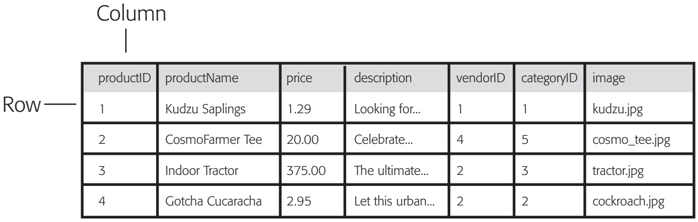
Simply put, databases store information. You encounter them every day in one way or another, whether charging a dinner on a credit card or calling Moviefone to get local movie listings.

A database is like an electronic filing cabinet that stores related information. At home, you might have a filing cabinet to store the bits and pieces of your life. You might have a filing folder labeled Insurance, in which you keep information about the various insurance policies you carry. Other folders might contain information on phone bills, car service records, and so on. Databases work more or less the same way, as the following sections explain.

Tables and Records

Databases have an electronic equivalent to filing folders: tables. A *table* is a container that holds information about a set of similar items. In the *CosmoFarmer* online store database, one table stores information on all the products for sale on the site.

This “Products” table tracks certain information—the name of the product for sale, its price, a short description, and a few other items. Each piece of information, like price, is stored in a column. All the information for each product (all the columns taken together, in other words) makes up a single *record*, which is stored in a row (see Figure 22-13).



productID	productName	price	description	vendorID	categoryID	image
1	Kudzu Saplings	1.29	Looking for...	1	1	kudzu.jpg
2	CosmoFarmer Tee	20.00	Celebrate...	4	5	cosmo_tee.jpg
3	Indoor Tractor	375.00	The ultimate...	2	3	tractor.jpg
4	Gotcha Cucaracha	2.95	Let this urban...	2	2	cockroach.jpg

Figure 22-13:
This diagram shows part of the Products table’s structure, and information for four records. Each row in a table represents a single record, or item, while each piece of information for a record is stored in a single field, or cell.

If you were designing a database, you’d try to model a table on some real-world item you needed to track. If your database was used for generating invoices for your business, you might have a table called Invoices in which you’d store information such as the invoice number, date, and so on. Since your customers are another source of data that needs tracking, you’d also create a table called Customers to store the information about them.

Tip: If you’re designing a database to track a business process that you already track on paper, a good place to start is with the paper forms you use. If your company uses a personnel form for collecting information on each employee in the business, you’ve got a ready-made database table. Each box on the form is the equivalent of a category column in a table.

In addition to the Products table, CosmoFarmer also tracks the vendors who manufactured the products. (After all, after they run out of inventory, the online store staff will need to order more products from their vendors.) Because a product and a vendor are really two different things, the database has a *second* table, called Vendors, that lists all the companies that make the products for sale on the Web site.

Note: Some databases are extremely picky about the names you give your tables and database columns (also called fields). For example, you can't have a database column named Date in an Access database.

You might think, "Hey, let's just put all that information into a single table." After all, you could consider the vendor's information part of the information for each product. While it seems like this method might simplify things (because you'd have one table instead of two), it can actually create a lot of problems.

Imagine a scenario where CosmoFarmer stores both product and vendor information in a single table: CosmoFarmer begins selling a hot new item, *Kudzu seeds*, from Seeds R' Us. All the product information, including the name and price of the plant, as well as the phone number and mailing address for Seeds R' Us, are stored in a single table row. Next month, Seeds R' Us offers another new product, Eucalyptus Saplings, as part of its "invasive plant of the month club." But, in the meantime, Seeds R' Us has moved locations and changed its phone number. So when someone at CosmoFarmer adds the new plant to the Products database, she adds the new phone number and address as part of the new plant's record.

Now the database contains *two* sets of contact information for Seeds R' Us—one for each plant. Not only does this redundant data take up extra space, but the contact information in one record is now wrong.

You could run into an even worse problem when deleting a record. Suppose that the online store decides to discontinue the two plants from Seeds R' Us. If a *CosmoFarmer* staffer removes those two records from the database, she also deletes any contact information for Seeds R' Us. If the *CosmoFarmer* staff ever decide to stock up on kudzu again, they have no way of contacting the vendor.

So you can see why it's prudent to keep separate classes of information in different tables. With two tables, when Seeds R' Us moves, you have to update only the information in the Vendors table, without touching the Products table at all. This way, if the staff deletes a product, they still have a way of contacting the vendor to learn about new products.

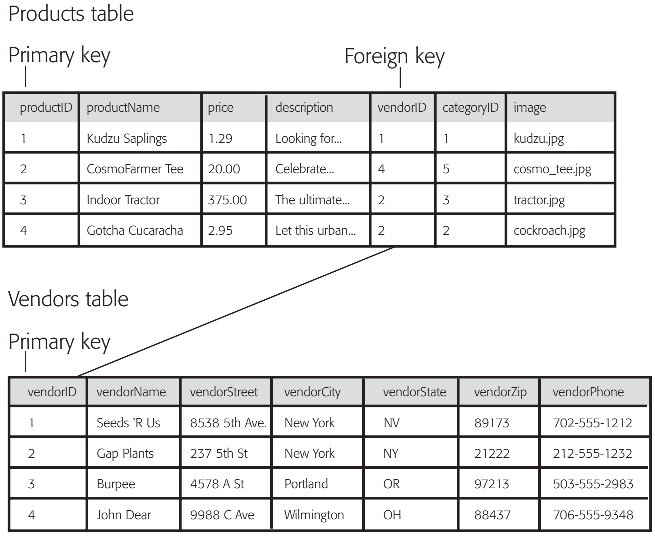
You may be wondering, with a setup like this, how to tell which vendor makes which product. All you have are two distinct tables—one with just product information and one with just vendor information. How do you make the connection?

Tip: For a great book on database design, check out *Database Design for Mere Mortals* (Addison-Wesley Professional) by Michael J. Hernandez. For concise, online introductions to database design, visit *www.geekgirls.com/databases_from_scratch_1.htm* and *www.campus.ncl.ac.uk/databases/design/design.html*.

Relational Databases

To connect information between tables, you create a *relationship* between them. In fact, databases that use multiple related tables are called *relational databases*.

The most common way to connect tables is by using what's called a *primary key*—a serial number or some other unique identifying flag for each record in the table. In the case of the *CosmoFarmer* database, the Products table includes a field named *productID*, the product's identification number (see Figure 22-14). Whenever a product is added to the database, it's assigned a new number (usually by the database server itself). If you're building a database that contains a table about employees, you might use an employee's Social Security number as a primary key, or an internal employee ID number based on your company's own cryptic method of identifying employees.



productID	productName	price	description	vendorID	categoryID	image
1	Kudzu Saplings	1.29	Looking for...	1	1	kudzu.jpg
2	CosmoFarmer Tee	20.00	Celebrate...	4	5	cosmo_tee.jpg
3	Indoor Tractor	375.00	The ultimate...	2	3	tractor.jpg
4	Gotcha Cucaracha	2.95	Let this urban...	2	2	cockroach.jpg

vendorID	vendorName	vendorStreet	vendorCity	vendorState	vendorZip	vendorPhone
1	Seeds 'R Us	8538 5th Ave.	New York	NV	89173	702-555-1212
2	Gap Plants	237 5th St	New York	NY	21222	212-555-1232
3	Burpee	4578 A St	Portland	OR	97213	503-555-2983
4	John Dear	9988 C Ave	Wilmington	OH	88437	706-555-9348

Figure 22-14:
Each table in your database should have a primary key–a column that contains a unique identifier for each record in a table. To relate information from one table to another, people often add an additional column with information pertaining to another table. In this case, a column called vendorID in the Products table contains a primary key from the Vendors table. To determine which vendor distributes, say, the Gotcha Cucarache, look at the fifth column in the Products table, which identifies the vendor's ID number as 2. When you check the Vendors table, you see that vendor 2 is Gap Plants. A column that contains the primary key of another table is called a foreign key.

The Vendors table has a primary key named, not so creatively, *vendorID*. This key is generated automatically whenever a new vendor is added to the database.

To join these two tables, you'd add another column called vendorID to the Products table (see Figure 22-14). Instead of storing *all* the contact information for a vendor within the Products table, you simply store the vendor's ID number. To find out which vendor makes which product, you can look up the product in the Products table, find the vendor's ID number in the vendorID column, and use *that* information to look up the vendor in the Vendors table.

While this hopscotch approach of accessing database tables is a bit confusing at first, it has many benefits. Not only does it prevent the kinds of errors mentioned earlier, it also simplifies the process of adding a new product from a vendor. When Seeds R' Us adds a third plant to their collection, a store staff person determines whether any of Seeds R' Us' info has changed (by checking it against the Vendors table). If not, she simply adds the information for the new product, and leaves the vendor's contact info untouched. Thus, relational databases not only prevent errors, they also make data entry faster.

Databases, of course, can be much more complicated than this simple example. It can take many tables to accurately hold the data needed to run a complex e-commerce site such as Amazon.com. In some cases, you may already be working with a previously created database, so you won't have to worry about creating one or even learning more than what's described above. For the tutorials in this section of the book, you'll use the already created *CosmoFarmer* database.

Loading a Database

You need to install the data for the CosmoFarmer store in your new MySQL server. That process requires a few steps. First you'll create a new database; next, you'll load the data into this new database; finally, you'll create a new *user* for the database (a special account that you'll use for accessing and updating the database). The following steps lead you through everything.

1. **In any Web browser, type *http://localhost/xampp* (*http://localhost/MAMP* if you're on a Mac).**

   This action takes you to the main XAMPP (or MAMP) page on the new testing server. You'll use a program called phpMyAdmin to administer the MySQL server.

2. **Click the phpMyAdmin link.**

   In XAMPP, this link is located in the left-hand list of links under the Tools category. The MAMP homepage lists phpMyAdmin in the top navigation bar.

Tip: Since you'll frequently use phpMyAdmin to work with the MySQL database, it's a good idea to bookmark this page, so you can quickly return to it whenever you need.

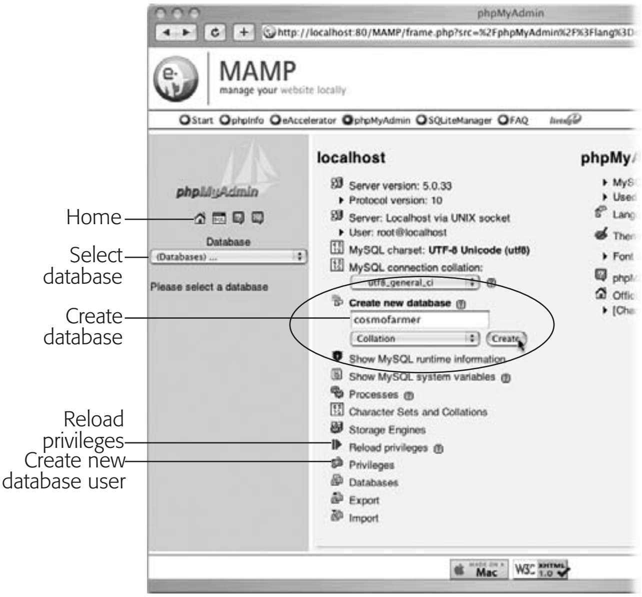
3. **In the "Create new database" box, type *cosmofarmer* and then press the Create button (see Figure 22-15).**

   This step actually creates a new database on the MySQL server. In addition, it takes you to a new page that includes a row of buttons for working with the new database.

   Next you'll load an SQL file that creates the required tables, and adds the actual data to the database. SQL stands for Structured Query Language, and it's the language you use to communicate with databases: to read, edit, update and generally manipulate the structure and information in a database. You'll learn more about how to use SQL on page 793.

4. **In phpMyAdmin's top navigation bar, click the Import button.**

   Doing so takes you to a page that lets you type in an SQL query (see page 793) or load a text file that has SQL commands in it. You'll do the latter—load a text file that contains all the SQL necessary to create the tables and data for the database.



localhost

Server version: 5.0.33
Protocol version: 10
Server: Localhost via UNIX socket
User: root@localhost
MySQL charset: UTF-8 Unicode (utf8)
MySQL connection collation: utf8_general_ci

Create new database: cosmofarmer

Please select a database

Figure 22-15:
The phpMyAdmin main page. Here you can create a new database, select an already created database to work with, and handle many administrative tasks for MySQL.

5. **Click the Browse button in XAMPP or the Choose File button in MAMP (circled in Figure 22-16). In the File Upload window that appears, navigate to and select the file *cosmofarmer.sql* in the *php_dynamic* folder you downloaded with the tutorial files.**

   As explained in the previous step, this file contains all the goodies that will appear in your database.

6. **Click the Go button (Figure 22-16, lower-right corner).**

   The MySQL server slurps down the SQL file, and executes the instructions found within it. The result? Four new tables are created, and a bunch of data is added to them. Your last step in prepping the database: Create a new MySQL user that has permission to add to and update the CosmoFarmer database.

7. **Click the link at the top of the page labeled *Server: Localhost*.**

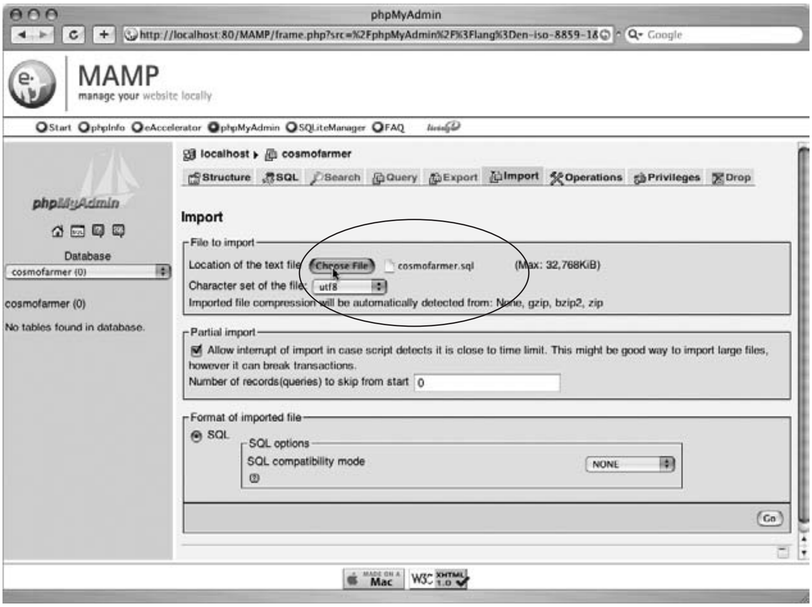
   Alternatively, you can click the small house icon in the left sidebar (see Figure 22-15). In either case, you return to the main phpMyAdmin page.

8. **In the middle column of links, click the Privileges link (see Figure 22-15).**

   This step opens the User page. From here, you can edit current users (for example, change a user password), and add new users.

9. **Click the "Add a new user" link.**

   The link appears under the list of current users. Clicking it opens the "Add a new User" page (see Figure 22-17).



Database: cosmofarmer (0)

cosmofarmer (0)

No tables found in database.

Import

File to import

Location of the text file: cosmofarmer.sql (Max: 32,768KiB)

Character set of the file: utf8

Imported file compression will be automatically detected from: None, gzip, bzip2, zip

Partial import

☑ Allow interrupt of import in case script detects it is close to time limit. This might be good way to import large files, however it can break transactions.

Number of records(queries) to skip from start: 0

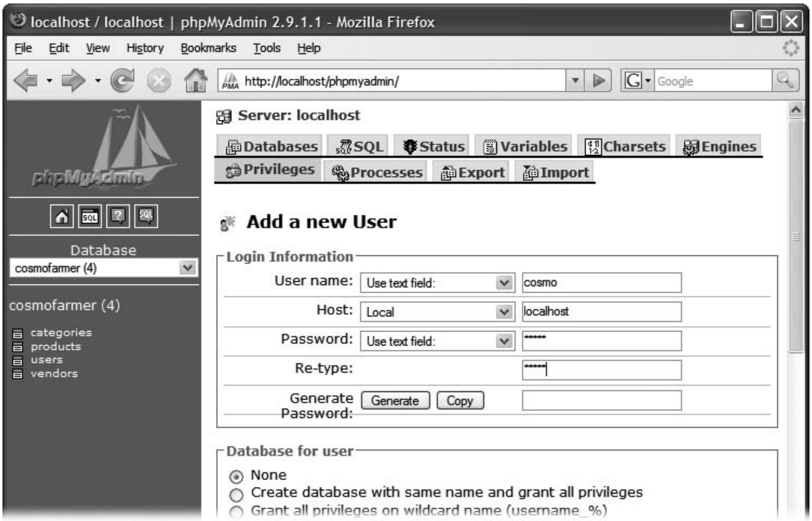
Format of imported file

● SQL

SQL options

SQL compatibility mode: NONE

Figure 22-16:
phpMyAdmin lets you load an SQL file, and execute the SQL code inside the file. Translation: Talk to the database server and tell it to create tables and fields, and add data to them. This way is great for replicating data from another database. In fact, phpMyAdmin can help you export all the tables and data from any database to which it has access.



Database: cosmofarmer (4)

cosmofarmer (4)

- categories
- products
- users
- vendors

Add a new User

Login Information

User name: Use text field: cosmo

Host: Local localhost

Password: Use text field: ******

Re-type: ******

Generate Password:

Database for user

● None

○ Create database with same name and grant all privileges

○ Grant all privileges on wildcard name (username_%)

Figure 22-17:
To create a new database user, just supply a name, choose where the user can access the database from (usually localhost), and then type a password.

10. **Type *cosmo* for the user name and *localhost* for the Host. In the password field, type *cosmo*, and in the re-type box, type *cosmo* again. The screen should look like Figure 22-17.**

    This step creates a user whose name is *cosmo,* whose password is *cosmo,* and who can access the database only locally from the server. This means someone out on the Internet can't try to log in to the MySQL server using the cosmo account; only local access—for example, PHP pages being run on the same computer—is allowed.

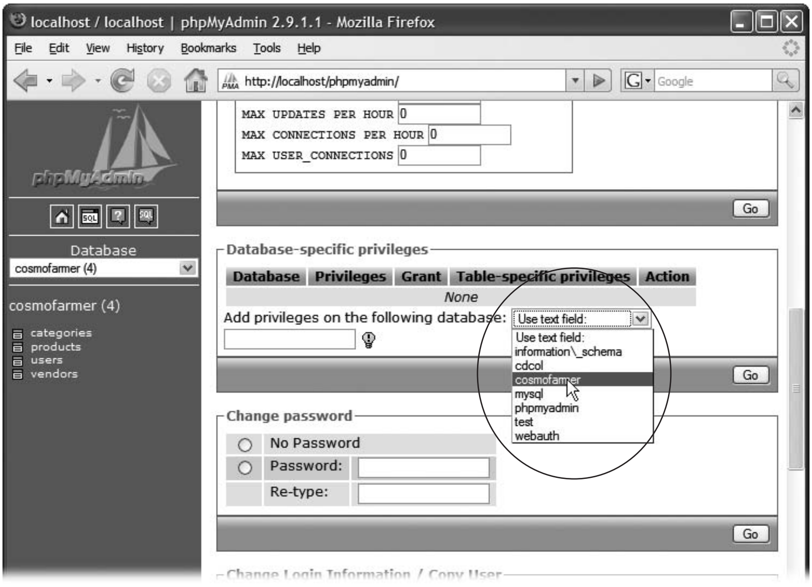
In general, it's a very bad idea to make a password the same as a user name, or to use any word that could be found in a dictionary, since it doesn't take much imagination for a hacker to figure this out and suddenly gain control of your database. But for this example application, it's best to keep things as simple as possible, so you can quickly get to the more interesting stuff (actually using Dreamweaver, for instance!).

11. **Scroll to the bottom of the page, and then, in the lower-right corner, click the Go button.**

    You've created the new user, cosmo, and now phpMyAdmin has taken you to another Web page so that you can tweak that user's privileges. In MySQL, you can limit which users have access to which databases, and how much power they have to work with the databases they do have access to. You could give a user the ability to read, update, and add data to a database, but prevent him from changing the structure of the database by adding or deleting tables. At this point, the cosmo user doesn't have any privileges, so you need to make sure cosmo can access the CosmoFarmer database.

12. **Scroll down the page to the "Database-specific privileges" section; select *cosmofarmer* from the database menu (circled in Figure 22-18).**

    This step takes you to yet another page, where you can specify what the user can and cannot do to the CosmoFarmer database.



localhost / localhost | phpMyAdmin 2.9.1.1 - Mozilla Firefox

http://localhost/phpmyadmin/

MAX UPDATES PER HOUR 0
MAX CONNECTIONS PER HOUR 0
MAX USER_CONNECTIONS 0

Database: cosmofarmer (4)

cosmofarmer (4)
- categories
- products
- users
- vendors

Database-specific privileges

Database	Privileges	Grant	Table-specific privileges	Action
None				

Add privileges on the following database: Use text field:

Change password

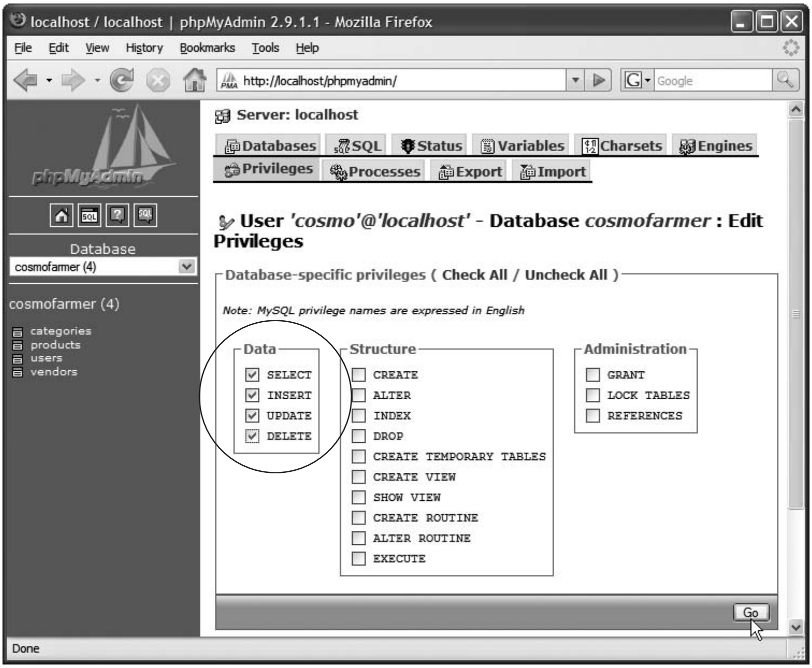
○ No Password
○ Password:
Re-type:

Figure 22-18: *The MySQL database server can have dozens, even hundreds, of databases running on it at the same time. Usually, only the main MySQL administrator account, called the root user, can access all the databases. Individual users, like the cosmo user you just created, are frequently allowed to access only a single database.*

13. **In the Data column, turn on all the checkboxes—Select, Insert, Update, and Delete (circled in Figure 22-19). In the lower-right corner, click the Go button.**

    You can give a user many different ways of interacting with a database, including some complex administrative functions that have the potential of wreaking havoc on the database. It's best to limit these "access privileges" to the minimum number each user needs. In this case, all the cosmo user needs to do is select (meaning retrieve data from the database), insert (put data into the database), update (edit data in the database), and delete (remove data from the database).

    There's just one final step in setting up this new user: You must "flush" the database's privileges. Although it sounds like a high school prank, flushing privileges just means telling MySQL to activate the new user and its privileges.



User *'cosmo'@'localhost'* - Database *cosmofarmer* : Edit Privileges

Database-specific privileges (Check All / Uncheck All)

Note: MySQL privilege names are expressed in English

Data
☑ SELECT
☑ INSERT
☑ UPDATE
☑ DELETE

Structure
□ CREATE
□ ALTER
□ INDEX
□ DROP
□ CREATE TEMPORARY TABLES
□ CREATE VIEW
□ SHOW VIEW
□ CREATE ROUTINE
□ ALTER ROUTINE
□ EXECUTE

Administration
□ GRANT
□ LOCK TABLES
□ REFERENCES

Figure 22-19:
This page lets you limit user access to just those functions the user needs for the task at hand. In this case, you use the cosmo user account in your PHP pages to access the database for routine database tasks such as adding, editing, deleting, and retrieving information from the database.

14. **On the left sidebar, click the Home icon to return to the main phpMyAdmin page; once there, click the "Reload privileges" link in the middle list of options (see Figure 22-15).**

    Alright, let's take stock of what you've done: set up the database, added the data, and created a new user. Wow. That was a lot of work, and you've barely touched Dreamweaver in this tutorial. Don't worry, all the hard work is behind you. Now it's time to use Dreamweaver to build the site.

Note: In many cases, when you set up an account with a Web hosting company, they supply you with an already created database and a user account for that database. Frequently, the database is named after your domain name or your account name with the Web host. When setting up MySQL on your machine as part of your testing environment, you should use the same user name and database name supplied by your Web hosting company. That way, when you've perfected your site on the testing server, you can simply transfer it to your Web host's server and the database connections should work perfectly.

Connecting to a Database

You've already defined a new site (back on page 762), so Dreamweaver knows that you'll be working on PHP pages, and knows the location the Web server's root folder. Now, you need to tell Dreamweaver how to connect to your newly created database. Fortunately, Dreamweaver makes this a snap.

1. **Return to Dreamweaver. From the Files Panel (Window → Files), double-click the file *index.php*.**

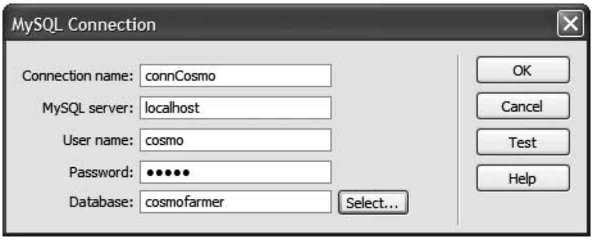
   The main page for the online store opens; this is a dynamic PHP page, which you need to have open in order to connect Dreamweaver to a MySQL database.

2. **Open the Databases panel by choosing Window → Databases.**

   The Application panel group opens. This is the control center for building dynamic Web pages.

3. **At the top left of the panel, click the plus sign (+) button. From the pop-up menu, choose MySQL connection.**

   The MySQL Connection window opens (see Figure 22-20). In this dialog box you let Dreamweaver know which database to connect to, where it's located, and the user name and password of the account that can access the database.



MySQL Connection

Connection name: connCosmo

MySQL server: localhost

User name: cosmo

Password: •••••

Database: cosmofarmer

Figure 22-20:
Setting up a Dreamweaver-to-MySQL-database connection requires just a few easy steps.

4. **In the "Connection name" box, type *connCosmo*.**

   You can use any name you want as long as it doesn't start with a number and doesn't contain any characters other than letters, numbers, and the underscore character. In this case, *conn* is a helpful indicator that this is a database connection, and makes identifying it easier if you ever need to look into the underlying code of the page.

Next, you'll tell Dreamweaver where the database is located.

5. **In the MySQL server box, type *localhost*.**

   In this case, both the Web server and MySQL are set up on the same computer (namely, yours). So when a dynamic page running on the testing server tries to connect to the MySQL database, the application server needs to look only on the same computer—"localhost"—to find the database.

   In many cases, the MySQL server provided by your Web hosting company is also located on the same server, so when you start building "real" PHP pages to put up on your own site, you can use *localhost* when creating the connection locally. This attribute is helpful, because it means you can develop your sites locally on your computer, move them onto the Internet, and they should be able to connect to the database without a problem.

   However, some Web hosts put their databases on separate machines dedicated to just that one task. In this case, the database connection on your local computer differs from the one you must put up on the Web server. You would need to edit the database connection, and replace *localhost*, which works for development and testing on your own computer, with the address of the MySQL server—this might be something like *mysql.webhost.com* or it might even be a basic IP address like *192.168.1.2* (don't use http://, or anything besides the server's address).

   Your Web hosting company can supply this information. Unfortunately this means that you need to change the connection file Dreamweaver creates to include the new address. Here's how: Before uploading your site, open the connection file. Do this either from the Databases panel (double-click the connection name to open it) or by looking in a folder name *Connections,* located in the local root folder of your site. In that folder you'll find a file named after the connection name you supplied when you created the connection—in this tutorial, that's step 4 above. So in this case the name of the connection file is *connCosmo.php*. Double-click this file, and then change *localhost* to the address of your Web host's MySQL database server. Then upload this file to your site (see page 620 for information on uploading Web files).

   You can then open the connection file again, and then change the database address back to *localhost.* This lets you continue to work and test on your local computer—just make sure you don't upload this file to the Web server, or you'll wipe out the connection file that you customized for your Web server, and your dynamic pages won't be able to connect to the database.

   This all assumes, of course, that the database and user name you set up on your testing server is the same as the one you use at your Web hosting company (see the Note on page 776). If not, you need to change the user name and password in the connection file before you transfer it to your Web server.

   Okay, with all *that* out of the way, back to our regularly scheduled programming: the MySQL Connection dialog box.

6. **Type *cosmo* in the user name box, and *cosmo* in the password box.**

This is the MySQL user name and password you created earlier when you set up the database.

7. **Click the Select button.**

   The Select Database window appears. This lets you pick which database you wish to connect to. In this case it's the CosmoFarmer online store database.

   If you get an error instead, check that you spelled *localhost* correctly, and that you supplied the right user name and password.

8. **Select *cosmofarmer*, and then click OK.**

   The dialog box closes, and *cosmofarmer* appears in the Database box at the bottom of the window. Click the Test button.

   A window saying "Connection was made successfully" should appear.

9. **Click OK to close the window that appeared when you tested the connection; click OK once more to close the MySQL Connection box.**

   Behind the scenes, Dreamweaver creates a small PHP file and stores it in a folder called Connections in your site's root folder. Whenever you create a dynamic page that communicates with the database, Dreamweaver adds a line of code pointing to this connection file. (The file's name reflects the connection name you typed in step 4—here it's *connCosmo.php*.)

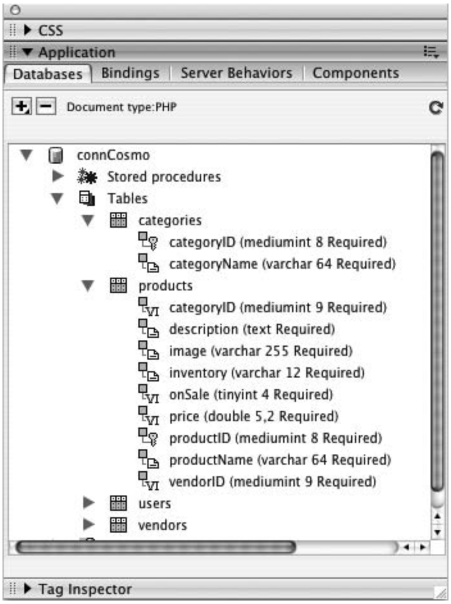
Note: Don't delete the Connections folder. This folder holds a script that lets your pages connect to your database. If, while cleaning your site, you throw this folder away, you'll break the database connection for all your site's dynamic pages. If this happens, recreate the connection by following steps 3–9.

Exploring the Databases Panel

The Databases panel (Figure 22-21) lets you do more than just connect databases to your site. It also lets you explore a database's structure and data. By clicking the + sign buttons (flippy triangles on Macs), you can view any of three lists:

- **Tables** lists all the tables in the database (see page 768). Expanding a table displays all the columns for that table. You'll use this option most often.
- **Views** lists all *views* stored in the database. A view is a selection of data in the database—a slice of its data. Unless you've created views using the database systems tools, this list is empty. Only version 5 and later of MySQL support this feature.
- **Stored Procedures** lists programs that access and manipulate information in the database. Stored procedures are kept right in the database, so they run faster than similar code in a Web page. (Some database systems—like Access and most versions of MySQL—don't recognize this feature.)

Tip: To get a quick peek at the data in a database table, in the Databases panel, right-click (Control-click) the table's name. From the shortcut menu, choose View Data. A window appears, displaying a table of data extracted directly from the database!



```
▼ connCosmo
    ▶ Stored procedures
    ▼ Tables
        ▼ categories
              categoryID (mediumint 8 Required)
              categoryName (varchar 64 Required)
        ▼ products
              categoryID (mediumint 9 Required)
              description (text Required)
              image (varchar 255 Required)
              inventory (varchar 12 Required)
              onSale (tinyint 4 Required)
              price (double 5,2 Required)
              productID (mediumint 8 Required)
              productName (varchar 64 Required)
              vendorID (mediumint 9 Required)
        ▶ users
        ▶ vendors
```

Figure 22-21:
The Application panel group contains four panels for working with dynamic database-driven Web sites. The Components tab contains advanced features for use with ColdFusion, JSP, and ASP.NET Web sites. (It doesn't have any effect on the ASP or PHP server models.)

In this chapter, you've laid the foundation for a dynamic Web site. In the next chapter, you'll start adding data from a database to the page you created in the preceding tutorial—and building a real, dynamic Web application.

FREQUENTLY ASKED QUESTION

Parenthetical Puzzler

In the Databases panel, I see some weird information in parentheses next to the column names–mediumint 8 Required, *for example. What's that about?*

You're right–there's a notation next to each column name. For example, Figure 22-21 shows a column called *categoryID*, which is followed by *(mediumint 8 Required).*

The information in parentheses denotes the *type* of data in that column. In this instance, it's an *integer* (a whole number like 1, 3, or 5), it's *8* bytes of data long (meaning it can be a very, very large number), and it's *required* (meaning that every new record *must* have a value stored in this field). Within each of these categories, there can be subtypes like time stamp, decimal number, and so on. Different databases recognize different data types, so there's quite a long list of possible data types for all the server models Dreamweaver supports.

These notations may appear cryptic, but they can come in handy. For example, if you're creating a form for updating or inserting a record in a database (as described in the next chapter), the data type and length can help you figure out what kind of information you're looking for and how long it should be, and help you when you're adding Spry Validation to your form (as described on page 422).

The categoryName column pictured in Figure 22-21 contains text (that's what "varchar" stands for) and is 64 characters long. It's also "required," meaning you can't add a record to the categories table and leave the categoryName field empty. So if you're creating a form to add records to this table, you want to create a required text field that accepts at most 64 characters (you learned how to do this back on page 409).

CHAPTER

23

Adding Dynamic Data to Your Pages

What sets a database apart from a mere pile of facts is its ability to selectively retrieve information. After all, when you visit Amazon.com, you don't want to see every piece of information on every single book and product they sell. You probably want to see just a list of books on a certain subject or by a particular author, and then view more detail about the books that pique your interest.

This chapter shows you how to use Dreamweaver to display database information on your Web pages. Because these concepts can be tricky, you may prefer to get some hands-on experience by completing the tutorial on page 821 before reading the rest of the chapter.

Retrieving Information

Since databases can contain lots of information, you need a way to find just the data that you want to display on a particular Web page. Even though your company keeps information about its products, customers, suppliers, and so on in one database, you may be interested only in, say, an alphabetical list of all your customers. After securing that list, you might want to display a particular customer's contact information, or perhaps the list of products that person bought.

Understanding Recordsets

To retrieve specific information from a database, you start by creating what's called a *recordset.* A recordset—also called a *database query*—is a command issued to a database asking for particular information: "Hey Database, show me all the customers listed in the Customers table." It's the heart of many database operations that you'll perform in Dreamweaver (and a piece of jargon you can't escape in the dynamic Web page business).

Recordsets let you retrieve specified columns in a database. They can also sort records alphabetically or in numerical order, as when viewing a list of products from least to most expensive. In addition, a recordset can zero in on a specific record based on information submitted by a visitor to the site or on information provided in a URL. In essence, recordsets let you winnow down massive amounts of database information in a fraction of a second—a powerful benefit, indeed.

Note for ASP.NET Users: Dreamweaver uses the term "DataSet" instead of "recordset" to refer to database queries in ASP.NET.

Creating Recordsets

Querying a database can be quite simple or extremely complex. Dreamweaver provides tools to get the novice database developer up and running quickly, while also supplying what's necessary to create more advanced recordsets. Whatever your level of expertise, you start by opening (or creating) a Web page to display database information on; then you open the Recordset dialog box using one of the following methods (each of which assumes you've set up a server model, as described in Chapter 22):

- On the Data tab of the Insert bar, click the Insert Recordset button (see Figure 23-1).

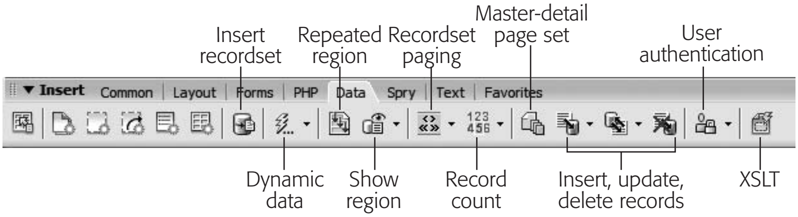
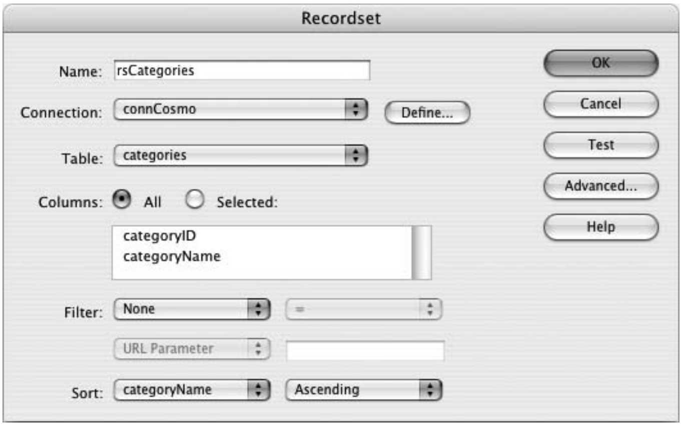


Figure 23-1: *The Data tab of the Insert bar provides one-click access to many powerful "application objects," which automate common dynamic Web page-building tasks. Actually, the first six buttons don't require dynamic Web pages: the first is for inserting data from a text file (described on page 266), while the next five are Spry Data objects (see page 476). (The Insert, Update, and Delete records buttons are discussed in the next chapter; the User Authentication features are discussed in Chapter 25; and the last option, XSL Transformation, is presented in Chapter 26.)*

- Choose Insert → Data Objects → Recordset.
- On either the Bindings or Server Behaviors panels in the Application panel group, click the + sign button (see Figure 23-9), and then select Recordset from the menu that appears.

Whichever technique you choose, the Recordset dialog box opens (Figure 23-2). This box lets you create a database query or recordset, and provides both simple and advanced modes of operation.



Name: rsCategories

Connection: connCosmo

Table: categories

Columns: ● All ○ Selected:

categoryID
categoryName

Filter: None =

URL Parameter

Sort: categoryName Ascending

Figure 23-2: *The Recordset window lets you retrieve data from a database. The main window (pictured here) lets beginners search and sort databases for specific information. Advanced options let seasoned database programmers take advantage of Dreamweaver's dynamic page-building abilities.*

To create a simple query, make sure you're in the *Simple* mode. (If a button labeled Simple appears at the right edge of the dialog box, click it to make it say Advanced. Now you're in Simple mode.)

1. **In the Name field, type a name for the recordset.**

   You can use any name you want, as long as it doesn't start with a number and doesn't contain any characters other than letters, numbers, and underscores (_).

Tip: A common technique is to begin the name with *rs* (*rsProducts*, for example). The *rs* helps you identify the recordset when you're working in Code view.

2. **From the Connection menu, select a database connection.**

   The menu lists all of the database connections you've defined for the site. If you haven't yet created a connection, you can do so now by clicking Define and following the instructions for creating database connections on page 776.

3. **From the Table menu, select the table that'll supply the data.**

   Information in a relational database is usually distributed among different tables, each of which holds information about a particular type of item, such as customer data or product data (see page 769). For example, to get a list of customers from a database, you'd select the Customers table (or whatever its name happens to be).

Note: To retrieve data from more than one table at a time, you need to create an *advanced* recordset (see page 793).

4. **To select columns from which you want to extract data, click the All or Selected button. If you choose Selected, then click the columns you wish to select.**

   By default, Dreamweaver highlights the All button, but you may not want to get data from *all* the columns in your table. For example, suppose your table contains lots of detailed information for each product your company sells. You may want to create a basic listing of all your products that simply displays names, prices, and descriptions. For this list, you don't need all of the details like SKU number, sizes, inventory status, and so on. Therefore, just select the three columns you're interested in.

   To select multiple columns, Ctrl-click (⌘-click) their names in the list in the Recordset dialog box.

   It's always best to limit your recordset to just those columns whose information you need. The more data you retrieve, the more you make the application and database servers work, and the more you slow down your site, especially when the database is large.

5. **Choose a Filter option, if you like.**

   In many cases, you don't want to retrieve *every* record in a table. For example, if you're looking for a phone number of a particular customer in your database, you don't want the details on every one of your customers. *Filters* let you limit the records retrieved by a recordset. (Details on the next page.)

6. **Choose a Sort option, if desired.**

   Data from a database may not appear in any particular order. Dreamweaver's sort options let you sort information based on a particular column. For example, maybe you're creating a recordset that gathers the title and release date for every CD you own. You might want to sort the results in alphabetical order by the title of the album, or chronologically by the date they were released.

   To sort database records, choose a column to sort by from the first Sort menu (Figure 23-2). Then select the sort order: either Ascending (A–Z, 0–10, earliest to latest) or Descending (Z–A, 10–0, latest to earliest).

   The Simple recordset mode lets you sort by only one column. That means, to continue with the previous example, if you want to sort records by date (so the most recent CDs appear first) and *then* by name (so CDs with the same date are then listed in alphabetical order), you have to use the Advanced mode (see page 793).

   To view the results of the recordset, click Test to open the Test SQL Statement window, which contains all records that match that query. If there are more than 25 matches, you can see the next group of results by clicking Next 25 at the bottom of the window. When you're done looking at the test results, click OK to return to the Recordset window.

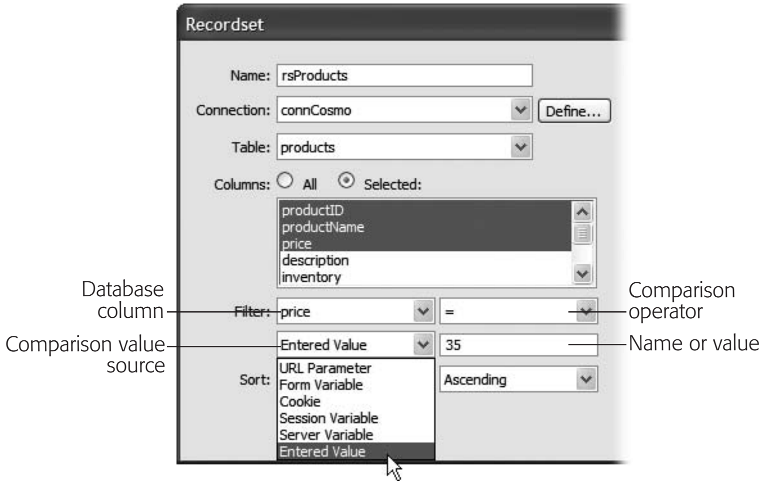
If the test results look right, click OK to close the Recordset window and add the code into the currently opened page.

Note: Unlike a database connection, which is listed in the Databases panel and is available to every page on the site, a recordset is specific to a particular page. (See page 798 to learn how to reuse recordsets on other pages.)

Filtering Information

Although you may have selected a limited number of columns when creating a basic recordset, the final results of the recordset still include *all* of the records within the table. That's fine when you want a list of all items in a database, like when you're creating a list of all your company's products. But it's not so useful when you want a particular subset of those records, like a list of just the red toupees your company sells, or when you want details on a *single* record—the "Flaming Inferno 78B toupee," for example.

To have Dreamweaver cull specific records from a table, use the Filter option in the Recordset window (see Figure 23-3). A *filter* lets you compare the information in one database column with a particular value and then select records that match—in other words, whenever you apply a filter to a recordset you're simply searching through the database for particular records. Suppose, for example, that your products database table contains a column named *price* that contains a product's price. To find all products that cost less than $35, you'd create a filter that looks for all records where the price column holds a value of less than 35.



Recordset

Name: rsProducts

Connection: connCosmo

Table: products

Columns: ○ All ● Selected:

productID
productName
price
description
inventory

Filter: price =

Entered Value 35

Sort: Ascending

URL Parameter
Form Variable
Cookie
Session Variable
Server Variable
Entered Value

Figure 23-3:
Filters let you limit the number of records retrieved by a recordset. Dreamweaver gives you three ways to apply filters to your records: information supplied in a URL; via a form submitted by a visitor to your site; or simply based on what you type into the recordset. Using a filter, a recordset can identify and retrieve data for a single record in the database, or a collection of records.

Using the Filter feature in the Recordset dialog box takes only a few steps:

1. **Create a recordset as described on page 782.**

   To create a filter, you must fill out the four form fields of the Recordset window's Filter options—three menus and one text field.

2. **From the first Filter menu, select a column name.**

   This is the column from the database that Dreamweaver compares to a particular value. In the previous example above, you'd select "price" from the menu to indicate the table's price column (see Figure 23-3).

3. **From the next menu, choose a comparison operator (< or >, for example).**

   To find products whose prices are less than $35, for example, you'd use the < (less than) operator. To find an exact value (all products that are exactly $35), use the = sign. Comparison operators are described below.

4. **Using the third Filter pop-up menu, select a source for the comparison value.**

   A filter compares the information in a table column against some other value. There are many different sources for such a comparison value. For example, on a Search page, you could create a form that allows visitors to type in a search term and click a Search button. In this case, the comparison value would come from a form. To set up this arrangement, you, the designer, would select Form Variable from this menu.

   For complete information on selecting a source for a comparison value, see the section "Getting Comparison Values" on the next page.

5. **Into the lower-right Filter box, type a name or value.**

   The value for this field depends on the source you selected in the last step; type in the name of the form variable, cookie, session variable, or whatever. The one exception: If you selected Entered Value in the previous step, type a specific value in this field. For instance, to compare the "price" column to a specific value, you'd select Entered Value and then type a number into the text field. The Recordset window would then look like Figure 23-3.

6. **Complete the Recordset window by choosing a sort option (if desired) and then clicking OK.**

   You can test the recordset and filter by clicking Test. If you selected anything other than Entered Value from the source menu, a message prompts you to type in a test value for the source—URL parameter, form variable, and so on.

Comparison Operators for Filters

Dreamweaver provides many different ways to compare information from a database column with information from another source, such as a form, cookie, or simply a value you type into the Recordset window. The type of comparison you choose also depends on the type of data you're comparing: text or numbers.

Comparing text values

You'll often want to create recordsets that find database matches to particular words. For example, a recordset could filter a list of products to find only those records whose descriptions contain the word "green," or you could filter a database of clients to search for a record for "Craig McCord."

Dreamweaver provides the following types of text comparisons:

- **Equality.** To check whether the information in a column is *exactly* the same as another value, select the = sign from the comparison menu.
- **Inequality.** To find records that don't match a particular piece of text, select the <> (doesn't match) operator from the menu. You would use this, say, if you wanted to search a database of clothing for items that do *not* match a particular phrase (like "winter" in the Season column).
- **Begins With, Ends With, and Contains.** The last three comparison operators are ideal for locating one or more words within text. For example, a database of movies might have a column containing a short review of each movie. To locate reviews that included the words "horrible acting," you could choose the Contains option, which will find any movie that includes the phrase "horrible acting" anywhere in its review.

  The Begins With and Ends With options are more selective. The former finds records only when the text at the very beginning of a particular record matches; the latter works only when the text appears at the end. You probably won't use these options very often, but they could come in handy if you wanted to search a database for people whose names are Bob or Bobby, but not Joe-Bob. In this example, you'd use the "Begins With" option, and use Bob as the comparison value.

The other comparison operators (<, >, <=, >=) aren't very useful for searching text in a database. They're intended for comparing numbers, as described next.

Comparing numbers

Filters are particularly useful for numbers: finding products that cost less than $35, albums that were released in 1967, products with more than 3,000 items in stock, and so on. If you've taken basic algebra, these options for comparing numbers should be familiar: = (equal to), <> (not equal to), < (less than), > (greater than), <= (less than or equal to), or >= (greater than or equal to).

Getting Comparison Values

By now it should be clear that the Filter option of the Recordset window lets you compare data from one column with some other value. But you're probably wondering where this "some other value" comes from. It depends on which option you selected from the third drop-down menu—the Comparison Value Source menu (see Figure 23-3).

The most straightforward option is the last item in the menu: Entered Value. After selecting it, you simply type the value into the field to the right of the menu. This could be a number, a letter, or one or more words. So, to create a recordset that will find a product whose price is more than $50, you'd select the price column, the > (greater than) comparison symbol, and the Entered Value source option, and then type *50* into the value field.

Unfortunately, this kind of recordset is rather limited. The comparison value you specify (50) is hardwired into the recordset, making it very inflexible. What if a visitor wanted to see products that cost more than $15, $30, or $100? No joy. This recordset is limited to what you, the designer, entered as a value.

You're better off creating the filter on the fly from information you get when the visitor's Web browser requests the recordset. In this way, you can create very flexible recordsets that are capable of searching databases for a variety of different pieces of information, not just the *one* value selected by a programmer. (After all, how good a search engine would Google be if the *programmers* determined what the search criteria were? No matter what you searched for—*Web design, Used cars*—it would always find Web sites about Java, Burning Man, and Diet Coke.)

Dreamweaver can also draw a filter value from a form, cookie, or even a link's URL. The process is always the same: From the filter's Comparison Value Source menu (Figure 23-3), select the source you want, and then type the name of the appropriate source item. For example, if you select Form Variable from the source menu, type the name of the form field in the box to the right.

In most cases, you must depend on an additional Web page to provide the source of these values. For example, a search function on a Web site usually requires more than a single page: one (or more) pages containing a Search field and a Submit button to send the information, and another that displays the results of the search. In this example, the form on one page sends information (the search terms) to another page (the results page), which uses the form information to search the database. In essence, Dreamweaver uses the words typed into the search form on one page to create the recordset on another page.

The two most common ways to pass information from one page to another are forms and URLs. (Three advanced sources—cookies, session variables, and application variables—are discussed on page 901.)

Form variables

A *form variable* is simply the information that a visitor types into a form field (or the value of a selected radio button, menu item, or checkbox). Forms are discussed in depth in Chapter 11, but their use in recordset filters is straightforward:

1. **Create a form page.**

   It can include a text field, pop-up menu, or some other form element. Make sure you *name* the form element. For use in a *simple* recordset filter, you're limited to a single form variable. Using an *advanced* recordset (see page 793), you can use information from more than one form field to filter the data in a recordset.

   If you wanted to give your site's visitors a chance to look at differently priced products, for example, you could create a menu that included the values 10, 50, 100, 500, and so on. People could then choose one of those options to look at products below the selected price. (Also be sure to give the menu a name, such as "price," as described on page 414.)

Tip: Name the form field the same as the database column you're filtering. For example, if you're searching a table's *productName* column, name the form field *productName*. If you do, Dreamweaver automatically fills out the correct form field name in step 6 below.

2. **Set the Action property of the form (see page 402).**

   You'll want it to point to the results page.

Note: For these steps to work, the form's method must be set to *Post* in the Property inspector (see page 403). If *Get* is selected, the form information appears in the URL when the form is submitted, and that information isn't available as a form variable. (You can, however, use the Get method in conjunction with the URL parameters option discussed next.)

3. **Open (or create) the results page.**

   This page displays the results of the recordset that's created using information from the form. This page needs to be a dynamic page using the server model you've chosen—ASP, PHP, and so on. (See page 766 for information on how to create a new dynamic page.)

4. **Add a recordset to the page, using the directions on page 782.**

   You'll also create a filter using a form variable.

5. **From the Filter menu, select a database column. Then choose a type of comparison, as described on page 785.**

   All of this is the standard filter-creation routine.

6. **From the source pop-up menu, select Form Variable. In the box to the right of the source menu, type the name of the form field that contains the value for comparison.**

   In keeping with the previous example, you'd type *price* into the box, since that's the name of the menu on the form page.

7. **Add a sort option, if you like, and then click OK to create the recordset.**

   Remember that this kind of recordset's results depend upon information sent from a form. If a visitor just stumbles across the results page without using a form, the recordset most likely produces no results (for a workaround to this problem see the box on page 791). That's why you should link to this kind of page only by using a form's *Action* property (see page 402).

URL parameters

In your Web travels, you've probably encountered URLs that look kind of strange, along the lines of *www.cosmofarmer.com/cart.php?productID=34&quantity=4*. Everything up to the *?* probably looks familiar, but you might be wondering what the *?productID=34&quantity=4* means.

Forms aren't the only way to pass information to a dynamic Web page; URLs can do it, too, thanks to information tidbits called *parameters*. Dynamic Web sites can read parameters and use them to create a recordset, among other things. (In fact, using the Get method for a form puts the form's information into the URL.)

You yourself can manually add a URL parameter to a link, but it's even more common to dynamically create a URL parameter from another recordset. This is a common technique when you want to link from a long list of items (such as a store's catalog of products) to a single, dynamic page that displays information about a single item (such as the name, description, price, and photograph of a single product).

When a URL parameter is sent along with a link to a dynamic page, that page can use the URL parameter as a filter to search a database. To identify a single record in a database, for instance, the URL could contain a number identifying the record's *primary key* (see page 769 for a definition). You'll find an example of this trick in the tutorial on page 818.

The steps for using URL parameters to filter recordsets are similar to those for form variables. You need two pages, one with a link containing the URL parameter and another containing the recordset.

Tip: It's possible to add a link with a URL parameter on the *same* page as the recordset. For example, you could have several text links like "Products under $10" and "Products under $100" that link to the same page but have different URL parameters.

Creating a link with a URL parameter

Dreamweaver provides several ways to create a link that contains a URL parameter. The simplest way is to first highlight the item you wish to turn into a link—usually text or a graphic. Then, in the Property inspector's link box, type the link followed by a ?, a parameter name, an =, and the value (for example: *products.php?category=7*).

However, you'll probably find it easier to browse for the file and let Dreamweaver write all the complex stuff. To do so, just follow these steps:

1. **Highlight the item you wish to turn into a link.**

   In other words, select a graphic or text on the page.

2. **Click the folder icon (browse button) on the Property inspector.**

   The Select File window appears. (For more on creating links, see Chapter 5.)

3. **Browse to and select the page containing the recordset.**

   This is the page that displays the results of the database search.

4. **Click the Parameters box in the lower-right corner of the Select File window.**

   The Parameters window appears (see Figure 23-4).

TROUBLESHOOTING MOMENT

The Default Value for a Filter Source

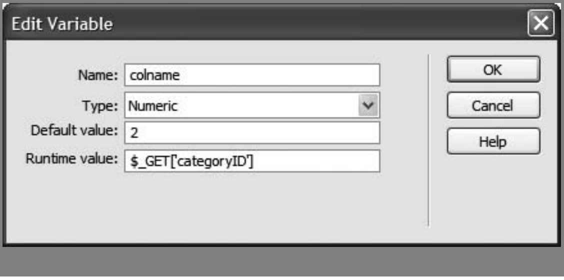
There's a problem with using a variable source of information for a filter. If the filter requires information from a form or URL parameter, what happens if someone comes to the page without first filling out the form or clicking a link with a URL parameter? In most cases, the recordset is empty, and the page displays no records. You can, however, set a *default* value for the form variable or URL parameter, so that at least some records always appear.

Using the steps outlined on page 782, create a basic recordset; include a filter using a form variable or URL parameter. Then click the Advanced button in the Recordset window.

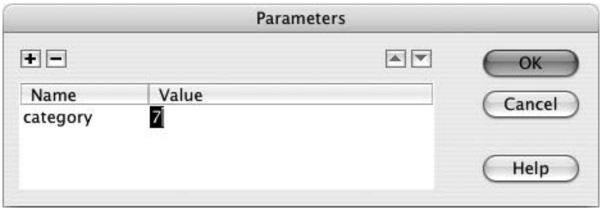
Now you get a more complex view of the recordset. In the Variables list, there's a single entry: *colname.* Select *colname* and click the Edit button to open the Edit Variable window. The "Default value" box indicates the value that will be used for filtering the recordset if none is supplied (in other words, when no form or url variable is provided by the Web browser). The value Dreamweaver supplies is *-1,* which probably won't match anything in your database. Change this value to something that will. For example, if you're filtering on a primary key (to identify one record in the database), just type a value that you KNOW exists in the primary key field for one particular record. For example, *1.*

You could type a value that matches *all* the records in the database. For example, if the recordset is used to find products under a certain price, type a value (price) that's larger than the most expensive product in the database. This way, the recordset retrieves all items under that price–in other words, all the products. (This trick also works for the other sources discussed on page 897: cookies, application variables, and session variables.)

One last word of warning. If you switch back to the basic recordset view by clicking the Simple button, Dreamweaver resets the recordset variable to the default value of *-1.* In other words, if you change the default value in the advanced view, *don't* switch back to the basic recordset view.



Name: colname
Type: Numeric
Default value: 2
Runtime value: $_GET['categoryID']



Name	Value
category	7

Figure 23-4:
The Parameters window lets you add URL parameters to a link. Recordsets can then use these pieces of information to filter a database query, as discussed on page 785.

5. **Click in the space below the Name column and type the name of the URL parameter.**

   What you're creating here is a name of your choosing that describes the data you're searching for. Avoid spaces and any punctuation characters for the name, since you're likely to run into troubles when you try to use such a name in the recordset filter. One approach is to use the name of the table column that you'll be filtering on when you create a recordset. For example, say you create a link that will lead to a page listing all of the products in a certain category (for instance, plants). Each product might have a column named *categoryID* which identifies the category the product belongs to. In this case, name the parameter *categoryID*.

6. **Click the space below the Value column and then type the value for the URL parameter.**

   This is the value that the filter in the recordset uses to match records in the database.

   Usually this is a simple value like *17, blue*, or *yes*. But you can also use spaces and punctuation marks in the value—for example, *Bob Jones*, in order to search for "Bob Jones" in the database. However, you need to make sure Dreamweaver's Preferences are set accordingly: choose Edit → Preferences (Dreamweaver → Preferences); click the Code Rewriting category and make sure the "Encode using &#" option is selected (the "Encode using %" option also works, but it can have trouble with some characters from some languages). Either of these options rewrites invalid characters in a form that works in a URL. For example, a space is converted to *%20.*

Note: Forms using the Post method don't suffer from any of these problems and can accept all types of punctuation and space characters.

7. **Click OK to close the Parameters window. Click OK to close the Select File window and apply the link.**

Creating the recordset for the results page

Once you've created the link, you need to create an appropriate recordset for the results page. Here's how:

1. **Open (or create) the results page.**

   This page displays the results of the recordset created using information from the URL parameter.

2. **Add a recordset to the page, using the directions on page 782.**

   You'll also create a filter using a URL parameter.

3. **From the Filter menu, select a database column. Choose a type of comparison, as described on page 786. From the source menu, select URL Parameter. In the box to the right of the source menu, type the name of the URL parameter.**

   This is the name supplied in step 5 of the previous instructions.

4. **Add a sort option, if you like; click OK to create the recordset.**

Like form variables, this recordset depends on information included in the URL of a link. If a visitor just stumbles across the results page without using a link with a URL parameter, the recordset most probably produces no results. Because of this, make sure you link to this kind of page only via a link with a parameter. Otherwise, modify the default value for the URL parameter in the recordset, as described in the box on page 791.

Tip: Using URL parameters (as opposed to form variables) for retrieving records has an added benefit: since the parameter is embedded in the URL–*http://www.cosmofarmer.com/products.php?productID=20,* for example–you can bookmark or email a link that matches a particular results page. In this way, you could bookmark, say, a page displaying all the products under $50. Then when you want to see if any new products (under $50) have been added to a site, you don't have to search the site again; merely revisit the bookmarked page.

Advanced Recordsets and SQL

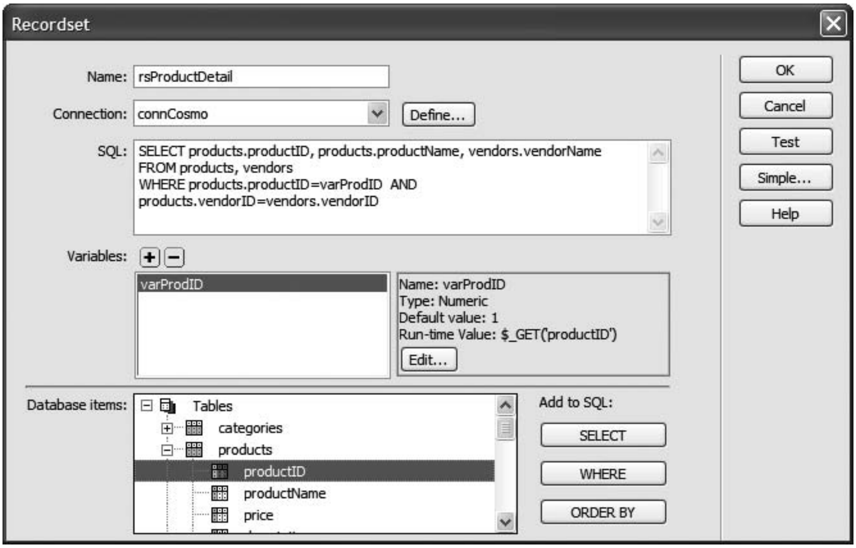
Sometimes you'll need more power than Dreamweaver's simple recordset tool gives you. For example, say you're building an online classified ads system. On one page, you want to present various pieces of information: the name of the sale item, its price, who's selling it, and how to contact the seller, for example. In order to store this kind of information, your database has two tables—one containing the ads themselves and one containing information about the sellers.

To present all this information, you need to simultaneously access both tables. In addition, you need to connect the records of those two tables so that each item for sale is associated with the correct seller—John Smith is selling the Whirligig 2007, for example. There's only one way to create this kind of complex query: using the advanced options of the Recordset window.

To display these options, insert a recordset using the steps described on page 782. Then, in the Recordset window, click the Advanced button. The Advanced Recordset window should appear (see Figure 23-5). (If you see a Simple button, then you're looking at the advanced options.)

Unfortunately, putting together advanced database queries is not as easy as most other operations in Dreamweaver. The Advanced Recordset window is basically just a way of typing in commands, using a database programming language called *SQL* (Structured Query Language, pronounced "ess-cue-ell"). SQL is a standard language that most database servers use to access, update, delete, and add information to a database.

To create an advanced recordset, type an SQL statement in the window's SQL box.



Recordset

Name: rsProductDetail

Connection: connCosmo

SQL:

```
SELECT products.productID, products.productName, vendors.vendorName
FROM products, vendors
WHERE products.productID=varProdID  AND
products.vendorID=vendors.vendorID
```

Variables: varProdID

Name: varProdID
Type: Numeric
Default value: 1
Run-time Value: $_GET('productID')

Database items: Tables, categories, products, productID, productName, price

Figure 23-5: *The Recordset window's advanced options aren't for the uninitiated. You need to have a good grasp of SQL–the standard database programming language–to make complex recordsets.*

SQL: the very basics

SQL lets you communicate with a database in order to find, add, update, and delete records. SQL even lets you do more advanced database work, such as adding new tables to a database and even deleting tables and databases. In the context of the Advanced Recordset window, you need to understand only how SQL *retrieves* information. After all, a recordset is just a selection of data pulled from the database.

To make an SQL query (called an SQL statement), you must first specify:

- **Which columns of data you want to retrieve.** For example, item price, item name, seller name, and seller contact information.
- **Which tables will supply this data.** In the earlier example, the information is stored in two tables: Ads and Sellers.
- **How the search should be limited.** You might just want products that are less than $10 or whose seller is Zachariah Smith. Or you might want to let visitors to your site make these choices themselves, via a form.
- **The sort order.** You could sort items using the Price column to view a list of items from least to most expensive, for example.

Only the first two pieces of information are absolutely necessary. A very basic SQL statement would look like this:

```
SELECT itemPrice, itemName
FROM ads
```

SELECT is an SQL keyword that specifies columns of data for retrieval; FROM indicates which database table contains them. This statement instructs the database server to look inside the Ads table and retrieve information from two columns: itemPrice and itemName. The result is a list of the price and the name of each item in the Ads table of the database.

UP TO SPEED

Getting Your Feet Wet in SQL

SQL isn't difficult to pick up. While you can create very complex SQL queries, a basic SQL statement is straightforward. Once you've reached the limits of Dreamweaver's basic recordset, you may want to expand your skills beyond this simple tool.

A great place to start learning how to write SQL statements is in Dreamweaver itself. After you create a simple recordset (see page 782), click the Advanced button. The SQL statement for the simple query appears in the SQL box.

This chapter introduces the very basics of SQL. For a more complete introduction, check out SQLCourse.com (*www.sqlcourse.com*). Or pick up a book like *SQL Queries for Mere Mortals* (Addison-Wesley Professional) by Michael Hernandez, *SQL in a Nutshell, 2nd Edition* (O'Reilly) by Kevin Kline, or *Sams Teach Yourself SQL in 10 Minutes* (Sams) by Ben Forta.

Note: SQL keywords are usually written in all caps—SELECT, for example. This is just a convention, not a hard-and-fast rule; "select" would also work. But since it's easier to identify the keywords if they're capitalized, it's best to stick with this convention.

Of course, you may not always want *every* record in a table. You may want to limit the search to a select number of items, such as products under $10. The WHERE keyword lets you do just that:

```
SELECT itemPrice, itemName
FROM ads
WHERE itemPrice < 10
```

Now the SQL statement retrieves only the price and the name of products that cost less than $10. Finally, SQL can sort records into order. In this example, you could also sort all the results from least to most expensive, like this:

```
SELECT itemPrice, itemName
FROM ads
WHERE itemPrice < 10
ORDER BY itemPrice ASC
```

The ORDER BY keywords indicate which column Dreamweaver should use to sort the records. Specifying the prodPrice column sorts the items by price. ASC is short for *ascending*, meaning that the records appear in low-to-high price order. (DESC sorts records into *descending* order, Z–A, or high-to-low.) You can even sort by multiple columns. If, for example, you wanted a list of all products sorted by price and *then* alphabetically by product name, you would simply change the above ORDER BY keyword to read like this:

```
ORDER BY prodPrice ASC, prodName ASC
```

In this way, all the products that were the same price (for example, $10) would then be presented in alphabetical order (A–Z).

Using the Data Tree view

Although you need to know SQL to use the Recordset window's advanced options, you can get a little help from the data tree in the "Database items" list at the bottom of the window (see Figure 23-5). This area of the window functions just like the Databases panel and lets you view the tables, columns, views, and stored procedures in the database (see page 778).

Click the + (arrow) button next to the word Tables to see a list of all tables in the database. Click the + (arrow) next to a table name to see all the columns within that table. This technique is very helpful when you're building an SQL statement, because you may not remember the exact names of every table and column in your database.

To build an SQL statement, you can select a column name and click one of the three buttons—SELECT, WHERE, or ORDER BY. The SQL command and column name then appear in the SQL box.

Suppose, for example, you wanted to create the following SQL statement:

```
SELECT productID, productName
FROM products
```

To build this statement using the data tree, click the + button next to the table named Products, which expands to show a list of all columns. Then click the column named productID and click SELECT. Next, click the productName column, and then click SELECT again.

Actually, when you use the Data Tree to insert SQL, Dreamweaver doesn't exactly write it the same as listed above. What you would get after following the previous instructions is:

```
SELECT products.productID, products.productName
FROM products
```

Dreamweaver inserts what are called "qualified" column names. For example, *products.productID* indicates the productID column from the products table. When you build a SQL query that involves more than one table, fully qualifying the column name by tacking on the table name as well is a good practice. If the same column name appears in both tables then your SQL won't work—for example, say the Products and the Vendors table both had a column called *name.* They each refer to different types of data—the item for sale, and the name of the vendor. Without differentiating the two columns (by "pointing to" them, for example as *products.name* and *vendors.name*) the database won't know which column you mean and you'll end up with a SQL error when you try to test the recordset.

Although these buttons can save you time, they don't check whether the SQL statement is valid. Unless you've got a decent grasp of SQL, you can easily create a statement that generates errors when the test server runs it.

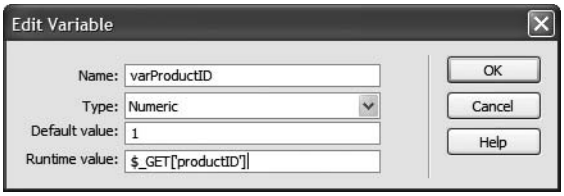
Creating variables for filtering data

Variables let you filter data using information from sources such as forms, URLs, cookies, session variables, and application variables. If you use the filtering option in the basic Recordset window, Dreamweaver creates a variable for you—but in the advanced Recordset window, you must create them yourself.

To add a variable for use in an SQL query, follow these steps:

1. **In the Recordset window, click the + button next to Variables (see Figure 23-5).**

   The Edit Variable window opens (see Figure 23-6).



Name: varProductID
Type: Numeric
Default value: 1
Runtime value: $_GET['productID']

Figure 23-6: *This dialog box lets you create variables to customize how an SQL statement works. You're not limited to just using variables in the filter part of an SQL statement (the WHERE clause) either. You can include a variable as any part of the statement. For example, one variable might determine the order of a sort operation: ASC or DESC.*

2. **In the Name box, type a name for the variable.**

   The name shouldn't include spaces or other punctuation marks.

Tip: As with database connections and recordsets, it's a good idea to add a prefix to the variable's name so you can more easily identify it in Code view. For example, you could begin the variable name with *var*–varPrice, for instance–just as you'd begin a recordset name with *rs* (rsProducts, for example).

3. **Choose a data type from the Type menu.**

   Your options are Numeric, Text, Date, and Double. If the variable is going to be used as part of a filtering operation, then the type you choose depends on the type of data stored in the column you're filtering on. For example, if the variable is used as part of a search on the name of a product, choose Text.

   The numeric and text options are, of course, for numbers and text. Use Date when the variable contains a date type. This isn't just any old date, however—it's the type of date defined by your database system. Each database has a different approach to this setting (to find out how MySQL handles date types visit http://dev.mysql.com/doc/refman/5.0/en/date-and-time-types.html). Finally, the Double option is reserved for numbers that are stored with two decimal places—mostly monetary values like 10.25 or 9.99.

Tip: You can easily determine the type of data stored in a column using the Databases panel, as described in the box on page 778.

4. **Type a value in the "Default value" box.**

   A default value comes in handy when the form, URL, cookie, session variable, or application variable is empty. The recordset uses the default value to filter the database records. See page 791 for more on when and why this might happen.

5. **Press Tab to jump to the "Runtime value" column; type the appropriate code.**

   The exact code depends on the server model you selected. For example, to retrieve the value of a form field named *price* you'd type *$_POST['price']* for PHP or *Request.Form("price")* for ASP. The best way to learn how to create variables is to use Dreamweaver's filter tool in the Recordset window (see instructions on page 785), and then switch to the advanced Recordset window, select the variable, and click the Edit button. The proper code for collecting information from forms, URLs, cookies, and so on appears in the variables' "Runtime value" field.

Note: Keep in mind that if you add more than one SQL variable in the advanced Recordset window, you can't switch back to the simple view.

Once you create a variable, you can include it in your SQL statement. Since variables help filter information, you'll often add them to the SQL *WHERE* keyword. For example, if you create a variable named *varPrice* that retrieves information from a form, you can add it to the SQL statement like this:

```
SELECT ads.itemPrice, ads.itemName
FROM ads
WHERE itemPrice < varPrice
```

In this example, whatever information is passed from the form is stored in the *varPrice* variable and compared to the price stored in the *itemPrice* column of the database.

If you ever need to edit a variable, just select its name from the Variables list in the advanced Recordset window and then click the Edit button. That reopens the Edit Variable window (Figure 23-6).

Reusing Recordsets

Recordsets are created on a page-by-page basis. In other words, when you create a recordset, it's added only to the current document. If you create another Web page that requires the same recordset, you must add the proper code to the new page. You can do this either by recreating the recordset—a potentially laborious process—or by simply copying the recordset from one page and pasting it into another.

Here's how:

1. **Open the Bindings panel by choosing Window → Bindings.**

   Ctrl+F10 (⌘-F10) also works. You can also copy and paste from the Server Behaviors panel.

2. **Right-click (Control-click) the name of the recordset you wish to copy; choose Copy from the shortcut menu that appears.**

   In the Server Behaviors panel, recordsets appear like this: *Recordset (rsName)*, with the name of the recordset inside the parentheses.

Now switch to the document that'll receive the pasted recordset. Right-click (Control-click) in the Bindings (or Server Behaviors) panel, and then choose Paste from the shortcut menu.

Tip: If you need a recordset that's similar to a recordset you've already created—but not identical—you can copy the original recordset, paste it into a new document, and then edit it, following the instructions in the next section.

Editing Recordsets

What if you have to edit a recordset? Maybe you forgot an important column of information when you originally created the recordset, or perhaps you want to modify a recordset you copied from another page. The process is easy: simply open either the Bindings panel (Ctrl+F10 [⌘-F10]) or Server Behaviors panel (Ctrl+F9 [⌘-F9]) and double-click the name of the recordset you wish to edit.

The Recordset window appears, looking just as it did when you first created the recordset (see Figure 23-2). Make any changes to the recordset, and then click OK.

Note: If you change the name of a recordset while editing it, Dreamweaver displays a message indicating that you need to use "Find and Replace" (see page 718) to locate and update every instance of the recordset's name. Dreamweaver opens the "Find and Replace" window for you when you click OK, but it's up to you to make sure the changes are correct.

This is another reason why beginning a recordset with "rs" (*rsProducts*, for example) is a good idea. If you've named a recordset simply "products," you could end up finding and replacing not only the name of the recordset, but also any other cases where the word "products" appears in the page.

The safest (although slowest) way to change a recordset's name is to recreate it. Of course, that's extra effort—a good argument for making sure you're satisfied with a recordset's name when you *first* create it.

Deleting Recordsets

If you add a recordset to a page and later realize that the page isn't using any of the information retrieved by the recordset, you should delete it. Each recordset forces the database server to do some work. Unnecessary recordsets only make your Web pages work harder and more slowly.

You can delete a recordset using either the Bindings or Server Behaviors panel. Just select the name of the recordset in either panel and click the minus sign (-) button at the top of the panel (pressing Delete on your keyboard has the same effect). However, if you've added dynamic data from that recordset to the page (described next) and you then delete the recordset, Dreamweaver won't remove references to the deleted recordset. In most cases, this breaks the functionality of the page and it won't work when viewed in a browser. You'll need to make sure you delete that dynamic information (as described on page 802) and remove any server behaviors that rely on that recordset from the Server Behavior panel.

Adding Dynamic Information

Once you've created a recordset, it's a snap to add dynamic information to a Web page. In fact, Dreamweaver provides several ways to add information. Start in the document window by clicking the spot where you wish to add the dynamic information. Then do one of the following:

- Choose Insert → Data Objects → Dynamic Data → Dynamic Text.
- Click the Dynamic Data button in the Data tab of the Insert bar and then select Dynamic Text from the menu (see Figure 23-7).

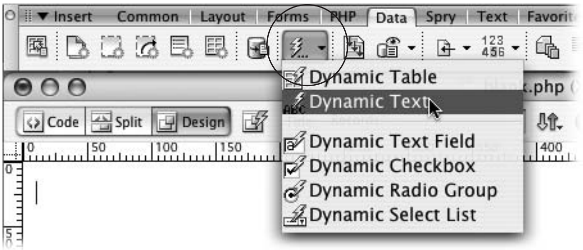
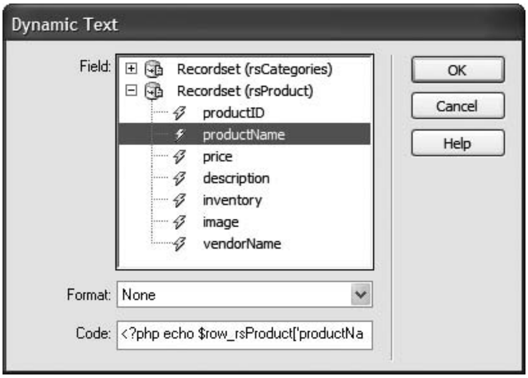


Figure 23-7: *The Dynamic Data button (circled) on the Insert bar's Data tab lets you add a variety of dynamic data to your Web page–from form fields filled in with information retrieved from a database, to a complete table based on a recordset.*

Either way, the Dynamic Text window appears (see Figure 23-8), listing the recordsets on the current page. Click the + sign button next to the recordset you wish to get information from. This expands to show all columns retrieved in that recordset. Pick the database column (also called Field) containing the information you wish to add to the page. You can pick only one column at a time, but you can repeat this process to add multiple columns to the page.

Note: *Dynamic Text* is a bit of a misnomer. This tool can also insert dates, numbers, and dollar values–not just text.

The format menu lets you format the data, like making a date appear as *January 17, 2008*. Formatting is discussed in depth in the "Formatting Dynamic Information" section below.



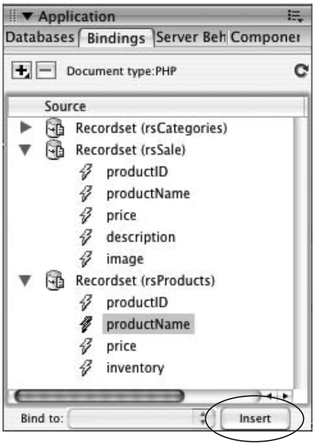
Format: None

Code: <?php echo $row_rsProduct['productNa

Figure 23-8:
After you select a database field to add to a page, Dreamweaver displays the necessary code in the Code box. This is the programming code that makes the data appear on your page. Dreamweaver writes this code using whichever programming language is compatible with the server model you've chosen.

The Bindings Panel

The Bindings panel (Figure 23-9) provides two other ways to put data from a recordset onto a page. (It's called Bindings because the panel provides a mechanism to "bind" or attach data from a database to a particular spot on a Web page.) Recordsets appear in the Bindings panel of the Application panel group. To open this panel, choose Window → Bindings or press Ctrl+F10 (⌘-F10).



Document type:PHP

Figure 23-9:
The Bindings panel lists all the different types of dynamic data you can add to your page. Recordsets (and the columns of data they've retrieved) appear here, but so do URL, Form, Cookie, and Session variables, as described on page 897.

To add data from the Bindings panel to a page, click in the document window where you wish to insert the dynamic data. Then select the column you wish to add in the Bindings panel, and click Insert (circled in Figure 23-9). But the best and fastest method is to just drag a column's name from the Bindings panel directly into the document window—into a paragraph or table cell, for example. You'll probably use this method most of the time.

After adding dynamic information to a page, it looks something like this: *{rsProducts. prodName}*. The information in braces indicates the name of the recordset (rsProducts) and the name of the data column (prodName). (You can make "real" data appear—instead of this code—when you use Dreamweaver's Live Data view, as described on page 812.)

Formatting Dynamic Information

Suppose a database includes a column for a product price and the raw information is listed in a pretty clunky format—something like *8* or *10.99*. But on your Web page, you want prices properly formatted with a dollar sign and two decimal places, like *$8.00* or *$10.99*.

Dreamweaver includes many different formatting options for numbers, dates, and text. You can choose a formatting option when using the Insert Dynamic Text window (Figure 23-8). You can also apply, remove, or change a formatting option using the Bindings panel.

To set a format using the Bindings panel (Ctrl+F10 [⌘-F10]), proceed as follows:

1. **In the document window, click the dynamic item you wish to format.**

   A down-pointing arrow appears under the Format column in the Bindings panel.

2. **In the Bindings panel, click the down-pointing arrow under the Format column and then select a formatting option from the menu (Figure 23-10).**

   Pick an option that's appropriate to the selected piece of data: Currency formatting for the price of a product, for example. If you try to apply a formatting option that isn't appropriate for the selected item (Currency to a text description, say), the page may produce an error when previewed on the testing server.

Tip for PHP users: Dreamweaver doesn't include date, currency, or number formatting options for the PHP/MySQL server model. Help is available in the form of a free extension called PHP Server Formats. Run, don't walk, to *www.tecnorama.org/document.php?id_doc=51* and download this extension now! For more information on what extensions are and how they work see page 738.

You can also format selected dynamic data just as you would other text on a page by applying CSS styles, headings, bulleted lists, and so on.

Deleting Dynamic Information

Dynamic information added to a page behaves like a single object. Just click to select it. You can also drag it to another location on the page, copy and paste it, remove it from a page by selecting it and pressing Delete, and so on.

If you remove all the dynamic information from a page, and don't plan on using any information from the recordset, make sure you delete the recordset, too, as described on page 799. Even though you may not display any dynamic information on the page, if a page contains a recordset, it must still communicate with the database server, slowing down your pages.

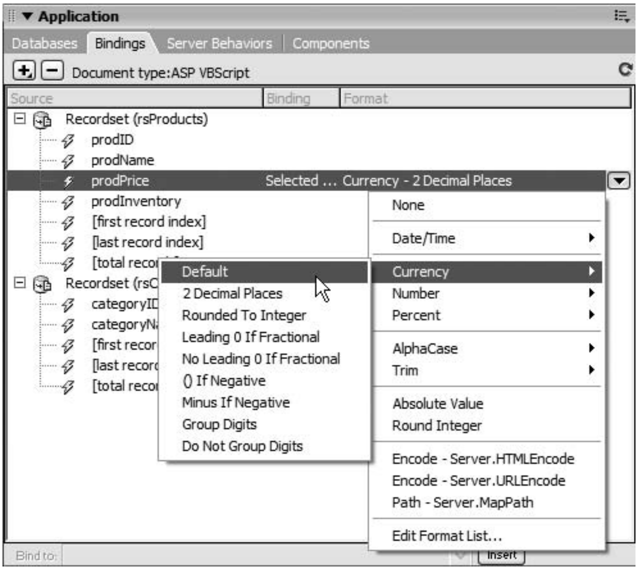


Figure 23-10:
The options shown here (for the ASP/VBScript server model) include a dizzying array of choices, including 19 date and time formats. One of them, "Encode – Server.HTMLEncode," may come in handy if your database stores HTML code (this option is called "Encode – HTMLEncode" in other server models). For example, maybe you've created a bulletin board whose messages are stored in a database. When you're adding one of those messages to a page, applying the "Encode – Server.HTMLEncode" format ensures that Web browsers don't render any code in a message as part of your page. Instead, any HTML in that dynamic text appears to visitors as code. Similarly, if there's any chance a data field might store a bracket like this <, the Encode formatting option prevents a Web browser from thinking that the opening bracket is the start of an HTML tag–a situation that could make your page not appear at all!

Dynamic data is also listed in the Server Behaviors panel—you can remove it just like you would any other server behavior, by selecting it from the Server Behaviors panel and then clicking the minus (-) button (see Figure 23-14).

Displaying Multiple Records

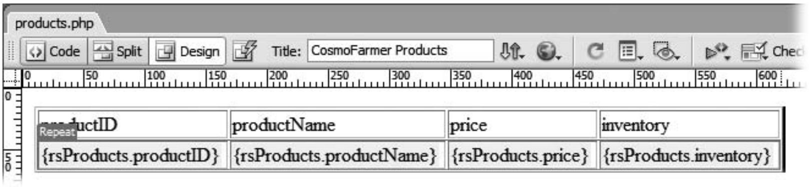
Often, you'll want to create a Web page that displays multiple records, such as a page that lists all the products in your company's catalog.

So far, the techniques you've learned for inserting dynamic data only insert information from a single record. Even if you've created a recordset that retrieves a thousand records, you'll still just see information from the very first record in the recordset when you simply drag data from the Bindings panel onto a dynamic Web page. There's a bit of extra programming that's required if you want to see information from more than one record (like a long list of all company employees) on a single page. Fortunately, Dreamweaver provides two tools for displaying multiple records: the Dynamic Table and Repeat Region objects.

Creating a Repeating Table

HTML tables (see Chapter 7) are meant to display data. The columns and rows of tables provide tidy compartments for individual pieces of information. It's not surprising, then, that in database terminology, *row* often refers to a single record in a database, and *column* indicates a single type of information in a record. Where a row and column meet, they form a "cell" that holds one piece of data from a single record.

Dreamweaver's Dynamic Table tool lets you display the results of a recordset in an HTML table. The top row of the table includes the name of each database column to identify the data in the rows below. The bottom row includes the actual dynamic data—one database column per table column (Figure 23-11).



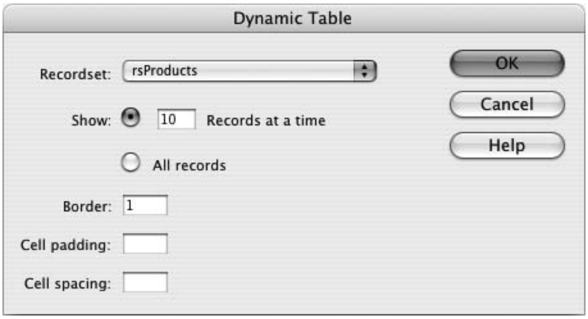
productID	productName	price	inventory
{rsProducts.productID}	{rsProducts.productName}	{rsProducts.price}	{rsProducts.inventory}

***Figure 23-11:** Although the Dynamic Table tool creates a table that displays multiple records, you see only one table row's worth of dynamic data. The Repeat tab that appears above the row indicates that this row will repeat, once for each record in the recordset. To see the effect, use Live Data view as described on page 812).*

The magic of this tool is that it can duplicate a row for each record in the recordset. In other words, the table displays multiple records, one per table row. Here's how to use it:

1. **Create a recordset (see page 782). In the Data tab of the Insert bar, select Dynamic Table from the Dynamic Data button menu** (see Figure 23-7).

   You can also choose Insert → Data Objects → Dynamic Data → Dynamic Table. Either way, the Dynamic Table window opens (see Figure 23-12).



Recordset: rsProducts

Show: ● 10 Records at a time

○ All records

Border: 1

Cell padding:

Cell spacing:

***Figure 23-12:** The Dynamic Table tool lets you quickly create a table to display all or some of the records in a recordset. This easy-to-use tool isn't available if you're using either of the ASP.NET server models. Instead, ASP folks need to use an ASP.NET datagrid–a more complex, but more versatile tool. For more information, check out www.adobe.com/devnet/dreamweaver/articles/data_grids.html.*

2. **From the Recordset menu, choose the recordset you'd like to use.**

   Since a page can contain more than one recordset, you have to indicate which records you want to display.

3. **Using the Show radio buttons, specify the number of records you wish to display.**

   You can show all of the records on a single page by clicking the All Records button. However, if your recordset is huge—if you're creating a Web page to display company employees and your company has 10,000 employees, say—a single page with all of that information could be very large and difficult to navigate. In this case, you should display only a handful of items at a time.

If you type *10* in this box, you see at most 10 records when previewing the page. (If you choose this method, make sure to add a Recordset Navigation bar, as described on page 807, to let visitors page through the list of results.)

4. **Set the Border, Cell Padding, and Cell Spacing values for the table.**

   These HTML table properties are described on page 254, All you're creating here is a plain old HTML table. You can dress it up later in Design view by changing the border, cell padding, cell spacing, background cell color, and other table properties.

Note: If you insert a repeating table and then later alter the recordset used in this table–for example, by adding a column of information to your query–Dreamweaver doesn't update the repeating table as a result of this change. In fact, if you remove a column of information from a query, you get an error when you try to preview the repeating table. If you edit the recordset and change the number of columns it retrieves (see page 799), it's usually easiest to delete the current repeating table and then replace it with a new repeating table.

5. **Click OK.**

   Dreamweaver inserts a table into the page. The top row contains the names of each column from the recordset—one name per table cell. You can, and probably should, edit these names to make them more understandable. After all, *productID* may not mean anything to your visitors.

   The bottom row of the table contains the dynamic data from the recordset and represents one record. Each table cell in that row simply holds dynamic text for each field in the recordset. You can select each of these placeholders and style them as you would any dynamic text—for example, using the Property inspector's style menu to apply a CSS style. The table row has a Repeat Region object applied to it, which you can edit or delete as described in the next section.

Creating a Repeat Region

While the Repeating Table is easy to use, it doesn't provide the flexibility you might require. For example, what if your dynamic information needs to be presented in a bulleted list (one record per bulleted item) or in individual paragraphs (one paragraph per record)? In these cases, you need to create a *repeating region*—a chunk of HTML that repeats once for each record in a recordset.

Here's an example that can help you understand how to use the Repeat Region object. Imagine you're creating a directory to list your company's staff. This page would include the name, telephone number, email address, and department of each employee. Of course, all of this information is stored in a table in a database—one column for each piece of information, and one record for each employee.

In Dreamweaver, you create the basic design of the page, and then add a recordset that retrieves the required information about each employee in the company. The page layout presents each employee's information in a single paragraph, so the finalized page has many paragraphs—one for each employee.

Since this is a dynamic page, you can't predict how many paragraphs you'll need. After all, your company may hire many new employees, which would add records to the database. To allow for this uncertainty, create a single paragraph by adding the dynamic information you want in it, following the steps on page 800. Then tell Dreamweaver to *repeat* that paragraph using information from each record the recordset retrieves.

Just follow these steps:

1. **Using the Bindings panel or one of the techniques described on page 800, insert dynamic text onto the page.**

   You should put these items together on the page, maybe in a single paragraph, in a bulleted or numbered list, or in a div tag (see page 313).

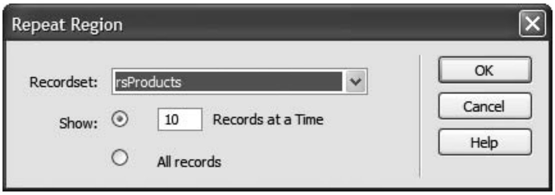
2. **Select the dynamic text and any HTML code you wish to repeat.**

   For example, select the paragraph (click the <p> in the document window's Tag selector) containing the dynamic data. If you're using a bulleted list to present this information, select the list item (<li> tag) containing the dynamic data. For data in a table row, select the table row (<tr> tag).

Note: You need to be very precise when selecting the HTML used in a repeating region. If you aren't, the page may not look or work as you expect. For example, if you select the dynamic data in a bulleted list, but don't actually select the <li> tag representing the list item, when you insert a repeating region, the records will repeat within a single bulleted list. In other words, you'll end up with a bunch of words (data from multiple records) as part of a single bullet. Use the Tag selector (see page 22) to accurately select an HTML tag.

3. **In the Data tab of the Insert bar, click the Repeated Region button (see Figure 23-1).**

   You can also choose Insert → Data Objects → Repeated Region. Either way, the Repeat Region window appears (Figure 23-13).



Recordset: rsProducts

Show: ● 10 Records at a Time

○ All records

Figure 23-13:
In a Repeated Region that reveals only a limited number of records at a time (in this case, 10), add a Recordset Navigation bar (see the next page). Otherwise, visitors to the page see the first 10 records of a recordset, but have no way to view additional records.

4. **From the Recordset menu, choose the recordset you want the page to work with.**

   Since it's possible to have more than one recordset per page, be sure to select the recordset whose data is included in the area you're going to repeat.

5. **Choose the number of records you wish to display.**

   If you decide not to use the "All records" option, make sure that you add a Recordset Navigation bar, as described below, to let visitors page through the list of results.

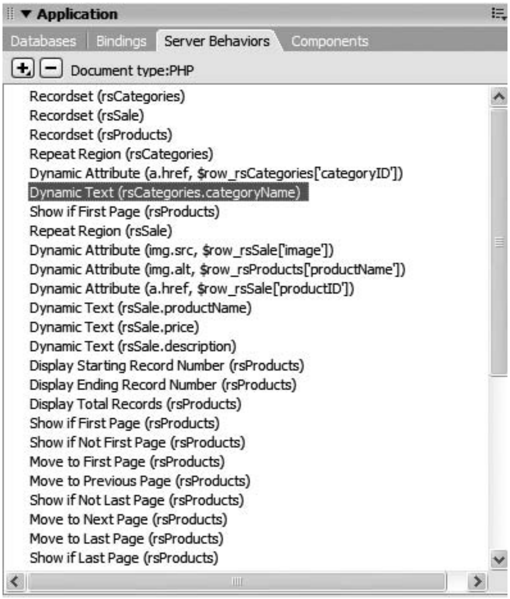
6. **Click OK.**

   Dreamweaver adds the proper programming code to the Web page.

Editing and Removing a Repeat Region

If you selected the wrong recordset, or you want to increase the number of records displayed at a time, you can easily change a repeating region. Simply open the Server Behaviors panel (Ctrl+F9 [⌘-F9] or Window → Server Behaviors) and double-click Repeat Region from the list. In the Repeat Region window (Figure 23-13), make any changes and then click OK.

To remove a Repeat Region, open the Server Behaviors panel (see Figure 23-14). Select Repeat Region from the list and click the minus sign (–) button (or press the Delete key).



Recordset (rsCategories)
Recordset (rsSale)
Recordset (rsProducts)
Repeat Region (rsCategories)
Dynamic Attribute (a.href, $row_rsCategories['categoryID'])
Dynamic Text (rsCategories.categoryName)
Show if First Page (rsProducts)
Repeat Region (rsSale)
Dynamic Attribute (img.src, $row_rsSale['image'])
Dynamic Attribute (img.alt, $row_rsProducts['productName'])
Dynamic Attribute (a.href, $row_rsSale['productID'])
Dynamic Text (rsSale.productName)
Dynamic Text (rsSale.price)
Dynamic Text (rsSale.description)
Display Starting Record Number (rsProducts)
Display Ending Record Number (rsProducts)
Display Total Records (rsProducts)
Show if First Page (rsProducts)
Show if Not First Page (rsProducts)
Move to First Page (rsProducts)
Move to Previous Page (rsProducts)
Show if Not Last Page (rsProducts)
Move to Next Page (rsProducts)
Move to Last Page (rsProducts)
Show if Last Page (rsProducts)

Figure 23-14:
The Server Behaviors panel lists every dynamic element you add to a page, from recordsets to repeat regions. Even when you drag a column from the Bindings panel to add dynamic data to a page, you create a server behavior. In this example, the highlighted element Dynamic Text (rs.Categories. categoryName) represents data from a recordset that will appear somewhere on the Web page. You can delete any server behavior by selecting it from the list and then clicking the minus (-) button.

Recordset Navigation

Dreamweaver's Repeating Region tool lets you display multiple database records on a single Web page. But a recordset with large amounts of data—like 1,000 employee records—can quickly choke a Web page. A large amount of information takes a long time to download and forces visitors to scroll for many screens to see it all.

Fortunately, the tool also lets you limit the number of records displayed at once. Of course, this limit presents its own set of problems: how do visitors see additional records, and how do you let them know where they are among all the records in the recordset?

To solve this dilemma, Dreamweaver comes with two handy commands for adding navigation to a recordset—and providing useful feedback about the recordset.

FREQUENTLY ASKED QUESTION

A Little Less Repetition

I applied the Repeat Region object, and when I preview the page, the same record is repeated over and over. That's not what I wanted to do! What's going on?

This can happen when you apply a Repeat Region object and inadvertently select the wrong recordset from the Repeat Region window (see Figure 23-13)–a mistake that's easy to make if you've included more than one recordset on a page.

The Repeat Region object adds programming code that steps through each record of a recordset. So, in practice, the Repeat Region object should get the information from the first record in a recordset and write it to the Web page, then go to the second record and add *its* info to the page. This process should continue until it's either gone through all the records in the recordset or reached the limit specified in the "Records at a Time" box.

However, if the information you want repeated is retrieved by a different recordset from the one selected in the Repeat Region window, this system breaks down. Instead, the code continues to cycle through each record from the selected recordset, but doesn't cycle through each record from the recordset containing the dynamic information you want repeated. The result: you get the same information over and over again. The first record of the recordset (the one you want repeated) is repeated for each record in the *other* recordset (the one you don't want repeated).

In other words, to ensure that Repeat Region works, select the recordset whose data is contained in the area you want *repeated*.

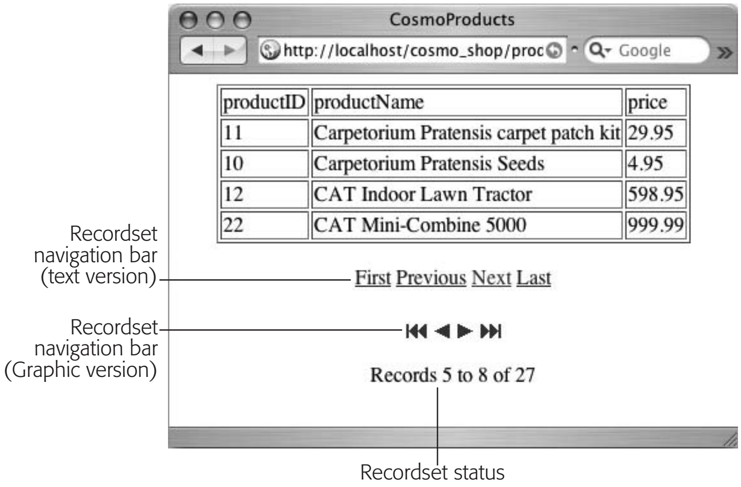
Recordset Navigation Bar

Suppose a page contains a recordset that retrieves 100 records from a database, but the repeating region on the page limits the display to 10 records at a time. In this case, you should insert a Recordset Navigation Bar to add either text links or graphic buttons to a page. These navigation bars let your audience view the next 10 records in the recordset, jump to the last records in the recordset (see Figure 23-15), jump back to the first record, or move to previous records in the recordset.

To add a Recordset Navigation bar, follow these steps:

1. **Click in the document window at the location where you want to insert the navigation bar. In the Data tab of the Insert bar, click the Recordset Paging button and then select Recordset Navigation Bar from the menu (see Figure 23-16).**

   You can also choose Insert → Data Objects → Recordset Paging → Recordset Navigation Bar. In either case, the Recordset Navigation Bar window appears.



CosmoProducts
http://localhost/cosmo_shop/proc
Google
Recordset navigation bar (text version)
Recordset navigation bar (Graphic version)
Recordset status

productID	productName	price
11	Carpetorium Pratensis carpet patch kit	29.95
10	Carpetorium Pratensis Seeds	4.95
12	CAT Indoor Lawn Tractor	598.95
22	CAT Mini-Combine 5000	999.99

First Previous Next Last

Records 5 to 8 of 27

Figure 23-15: *Dreamweaver's Recordset Navigation Bar object prevents information overload by letting Web page visitors step through easily digested pages of information. The Recordset Status message also keeps visitors informed about how many records are available.*

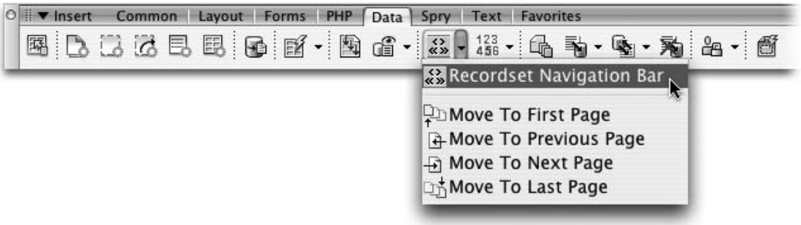


Figure 23-16: *The Recordset Paging button lets you insert a navigation bar for navigating the records returned in a recordset. In addition, if you want to build your own recordset navigation system, you can individually apply server behaviors like "Move to First Record" and "Move to Next Record". These server behaviors are discussed on page 906.*

2. **From the Recordset menu, select the recordset to navigate.**

   If the page contains more than one recordset, select the one that you used when you made the dynamic table or added the repeating region.

3. **Select whether to use text or graphic buttons for the navigation bar.**

   If you select Text, Dreamweaver proposes the words First, Previous, Next, and Last to indicate the navigation controls. (You can edit them later.) The graphic buttons resemble standard DVD player controls, representing forward and backward. If you select this option, Dreamweaver copies the four GIF files into the folder with the dynamic Web page. Later, you can replace these graphics with ones you create.

FREQUENTLY ASKED QUESTION

Behaviors That Serve You Well

What's the difference between Dreamweaver Behaviors (Chapter 13) and a server behavior?

Both are prewritten programs created by Dreamweaver's engineers. They differ mainly in where the programs run and what they attempt to accomplish.

A Dreamweaver Behavior is a JavaScript program that runs in a Web browser. It usually affects the interaction between a visitor and a Web page. For instance, the Swap Image behavior makes the Web browser exchange one image for another when a visitor mouses over a link. The behavior itself runs in the visitor's Web browser, and anyone can see the program by looking at the page's source code in the browser.

A server behavior, on the other hand, always runs on the *application server*—that is, on the Web-server side of the Internet. Instead of JavaScript, server behaviors can be written in a variety of different languages—VBScript, PHP, C#, Java, and so on, depending on the server model (page 751) your site uses. Server behaviors specifically let you create connections to databases and display, edit, and delete information from databases. Furthermore, since these programs run on the application server, your site's visitors never see the actual programming code. All they see if they look at the source of the page is plain-old HTML (the results of the server program).

In a nutshell, Dreamweaver Behaviors add interactive elements to a Web page, like rollovers and JavaScript alert boxes. Server behaviors supply the programming code you need to build sophisticated database-driven Web sites.

4. **Click OK to insert the navigation bar.**

   Dreamweaver inserts a table, consisting of one row and four columns, into the document window. Each cell contains one text or graphic navigation button. You can change the alignment and any other property of the table to fit your design (see Chapter 7 for more information on working with HTML tables).

Tip: If you use the Recordset Navigation bar frequently, you may long to replace the DVD-control graphics that Dreamweaver displays. Just create your own graphics and name them FIRST.gif, LAST.gif, PREVIOUS.gif, and NEXT.gif (make sure you capitalize them inconsistently just like Adobe has done). Place these graphic files in the C:\Program Files\Adobe\Adobe Dreamweaver CS3\configuration\Shared\UltraDev\Images folder (Applications → Adobe Dreamweaver CS3 → Configuration → Shared → UltraDev → Images folder.

Recordset Navigation Status

When you're viewing hundreds of records, it's nice to know where you are and how many records there are in all. The Recordset Navigation Status tool adds just such information to your pages, as shown in Figure 23-15. Dreamweaver presents the status message in the form of "Records 1 to 10 of 18," indicating which records the visitor is currently viewing and the total number of records.

Here's how to add a Recordset Navigation Status message:

1. **Click in the document window at the location you want to insert the status message. In the Application tab of the Insert bar, click the Record Count button (circled in Figure 23-17).**

You can also choose Insert → Data Objects → Display Record Count → Recordset Navigation Status. In either case, the Recordset Navigation Status window appears.

2. **Select a Recordset from the menu.**

   If the page contains more than one recordset, select the one that you used when you inserted the Recordset Navigation bar.

3. **Click OK to close the window and insert the status message.**

   The Recordset Navigation Status message is simply text with the three dynamic text items (see page 800 for more on dynamic text). Change the words "Records," "to," and "of" to anything you like, such as in "Products 1-10. 149 total products retrieved."

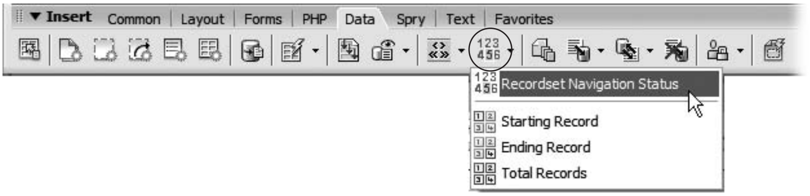


***Figure 23-17:** Use the Record Count menu to insert status information about a recordset, including the helpful message ("Records 1 to 10 of 18") provided by the Recordset Navigation Status server behavior.*

You can easily build you own recordset navigation status bar using the last three options in the Record Count menu in the Data tab (see Figure 23-17). Just click in the document window where you'd like to insert the recordset status information and then select an option from the Record Count menu.

The Starting Record option will display the number of the first record in a repeated region. The exact number will depend on where the visitor is within the recordset navigation. For example, say you added a repeated region that displays 10 records at a time; you then add a recordset navigation bar (page 807) so visitors can view all of the records in the recordset by paging through 10 records at a time. On the first page the starting record number is 1, but when someone clicks the "next page" button in the recordset navigation bar, the starting record on that page will be 11. In other words the starting record number is the "11" in "Showing records 11 to 20 of 100."

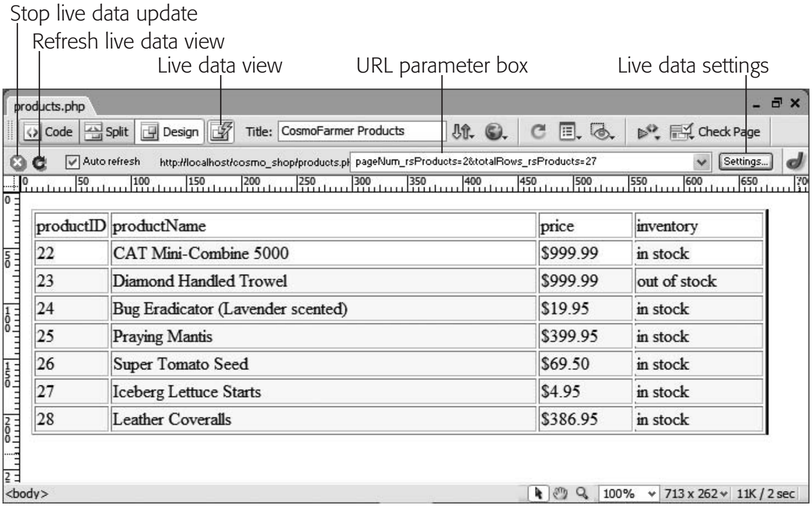
The same is true with the Ending Record option. It displays the number of the last record displayed on the page. Again, the exact value will depend on the recordset and which page of records is being viewed. The ending record number is the "20" in "Showing records 11 to 20 of 100."

Finally, the Total Records option is simple: it's just the total number of records retrieved in a recordset. This is a useful option that you'll find yourself using even without a recordset navigation bar. For example, say you wanted to know how many employees were listed in your database. Create a recordset that retrieves every employee (in other words, don't use a filter [page 785]) and then use the Total Records option from the Record Count menu.

Viewing Live Data

After you add dynamic information to a Web page, you see something like this in the document window: *{rsProducts.productID}*. That gives you an idea of what the information is—in this example, the database column productID from a recordset named rsProducts—but it doesn't show any real database information, which can make designing a page more difficult. You're especially far from seeing the actual result when a page contains a repeating region: what appears as a single row of dynamic text actually shows up as multiple rows or records when someone views it in a Web browser.

Fortunately, you can preview a page with real database records directly in Dreamweaver. In addition to Design view and Code view, Dreamweaver includes a view for working with dynamic information: Live Data view. When viewing a dynamic page in Live Data view, Dreamweaver contacts the testing server, retrieves recordset data from the database, and displays it in the document window, as shown in Figure 23-18.



productID	productName	price	inventory
22	CAT Mini-Combine 5000	$999.99	in stock
23	Diamond Handled Trowel	$999.99	out of stock
24	Bug Eradicator (Lavender scented)	$19.95	in stock
25	Praying Mantis	$399.95	in stock
26	Super Tomato Seed	$69.50	in stock
27	Iceberg Lettuce Starts	$4.95	in stock
28	Leather Coveralls	$386.95	in stock

Figure 23-18: *This is the Live Data view of the dynamic table pictured in Figure 23-11. In Live Data view, you can update the data displayed on the page by clicking the Refresh Live Data view button. Turn on "Auto refresh" if you want Dreamweaver to update the Live Data view automatically whenever you make a change to a dynamic element of the page (but avoid it if your connection to the testing server is slow). Also, if refreshing the data takes too long–or if when switching into Live Data view Dreamweaver seems to have stopped working–you can click the Stop Live Data Update button to halt the current update.*

To turn the Live Data view on or off, click the Live Data View button on the toolbar, choose View → Live Data, or press Ctrl+Shift+R (⌘-Shift-R). It may take a few seconds for the document window to change, since Dreamweaver must contact the testing server and retrieve information from the database. After a moment, the Live Data toolbar appears (see Figure 23-18), complete with tools for refreshing the displayed data, changing settings for the Live Data view, and adding URL parameters to test recordset filters.

Note: If you're using Microsoft's IIS Web Server and an Access database, you may get an error when you turn on Live Data view. The probable cause: a permissions problem that prevents IIS from controlling the database. You can find instructions for fixing this problem on Windows XP at *www.sawmac.com/missing/dw8/iis_trouble.php*.

With the Live Data view turned on, it's much easier to see what your page looks like when viewed on the Web. You can continue to work on a Web page in this view just as you would in Design view. You can add text and graphics, modify page properties, and even format dynamic data, as described on page 802.

However, when you're working with a repeating region, you can only select, delete, or format the *first set* of dynamic data items. For instance, as you can see in Figure 23-18, a dynamic table displays repeating rows of database records. If you wanted to apply a Cascading Style Sheet style to the name of each product listed, you would click the item in the first row of dynamic data—in this example, "CAT Mini-Combine 5000"—and apply the style to it. To see the style applied to the other records, click the Refresh button in the Live Data View toolbar.

Live Data View Settings

Some recordsets depend on information provided by a form or URL. Often when you use the filter option, for instance, a recordset searches a database for records that match information from a form or URL.

This feature can come in handy for pages that provide detailed information about a single record. Frequently, for these types of pages, the URL might appear something like this: *product.php?productID=38*, where the name of the page (*product.php*) is followed by a URL parameter that includes a name (*productID*) and value (*38*). The recordset then looks for the product whose ID (*productID*) matches 38.

Because pages like this can't show up properly without a little outside help, you need to provide extra information in the Live Data View Setting window, like this:

1. **In the Live Data toolbar, click the Settings button (see Figure 23-18).**

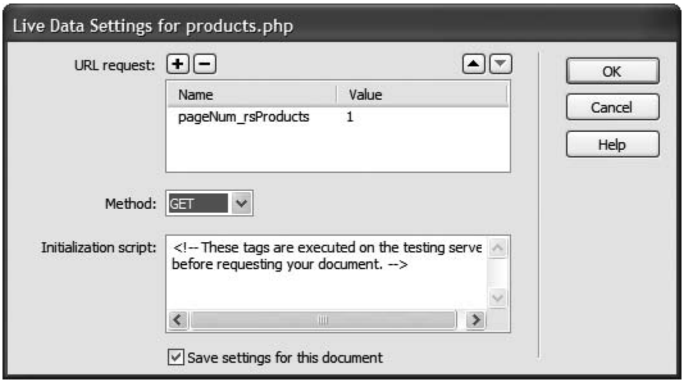
   You can also choose View → Live Data Settings. Either way, the Live Data Settings window appears (Figure 23-19).

2. **Click the + button to add a new name and value pair.**

   Dreamweaver refers to each name value pair in this window as a "URL request," but essentially it means either a form variable (see page 788) or a URL parameter (see page 789).

3. **Click the Name column and type a name for the new "URL request" item.**

   If the "URL request" is being used to filter data in a recordset, you would use the name you used when you created the filter in the Recordset window (see step 5 on page 786).



Live Data Settings for products.php

URL request:

Name	Value
pageNum_rsProducts	1

Method: GET

Initialization script: <!-- These tags are executed on the testing serve before requesting your document. -->

☑ Save settings for this document

Figure 23-19: *The Live Data Settings window lets you define information that the dynamic page needs to operate correctly. For example, the page may include a recordset that uses information contained in the URL to search records in a database. The Initialization Script section of the window stores temporary code that the application server executes before viewing the page in Live Data view. Use this advanced option for setting the session and application variables that the application server uses to process the page (see page 897).*

4. **Click the Value column and type a value.**

   This may be a number or text, but the value must retrieve at least one record from the database, according to the filter options you set up in the recordset. For example, if you created a filter to find products under a certain price, you might type *price* as the name of the URL request and *10* as its value.

5. **From the Method menu, select either GET or POST.**

   If the filter in your recordset uses a form variable, select POST; if the filter uses a URL parameter, select GET.

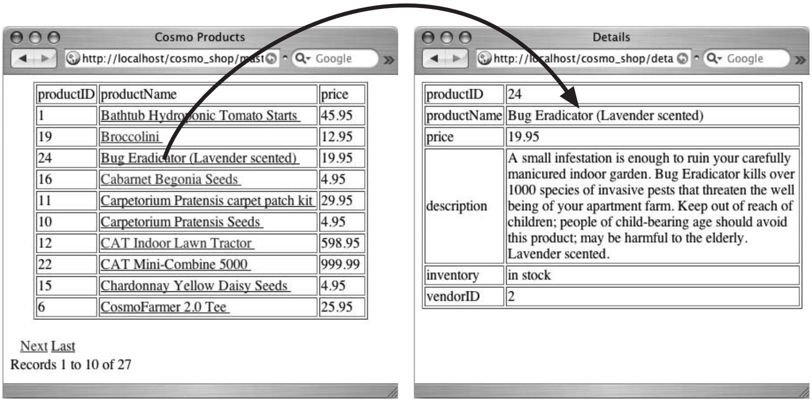
6. **Click OK to close the Live Data Settings window.**

   If you haven't turned on the Auto Refresh button, you must click the Refresh button in the Live Data toolbar (see Figure 23-18) to see the new results. In addition, if you selected the GET method in step 5, a URL Parameter box appears in the Live Data toolbar. You can change the values of the URL parameter directly in this box.

Note: The Recordset Navigation Bar and Status message objects react differently, depending on which records in a recordset are displayed. To see this effect in action, add a new URL request item named *pageNum_rsProducts* in the Live Data Settings window. Set the value to something other than 0. Click OK to return to the Live Data view. You can change this value directly in the URL parameter box in the Live Data View toolbar (see Figure 23-18) to see how the page reacts with different offset values.

Master Detail Page Set

When you build a database-driven Web site, you often want to give your visitors both an overview and a detailed view of information. Usually, it takes two separate Web pages to do the job: one that lists limited information about all the records in a recordset, and one that links to a second page with detailed information about a single record. Dreamweaver calls these *master* and *detail* pages and gives you a tool for making quick work of this task. Figure 23-20 shows you how these pages work together.


Cosmo Products
http://localhost/cosmo_shop/mast
Google
Details
http://localhost/cosmo_shop/deta
Google

productID	productName	price
1	Bathtub Hydroponic Tomato Starts	45.95
19	Broccolini	12.95
24	Bug Eradicator (Lavender scented)	19.95
16	Cabarnet Begonia Seeds	4.95
11	Carpetorium Pratensis carpet patch kit	29.95
10	Carpetorium Pratensis Seeds	4.95
12	CAT Indoor Lawn Tractor	598.95
22	CAT Mini-Combine 5000	999.99
15	Chardonnay Yellow Daisy Seeds	4.95
6	CosmoFarmer 2.0 Tee	25.95

Next Last
Records 1 to 10 of 27

productID	24
productName	Bug Eradicator (Lavender scented)
price	19.95
description	A small infestation is enough to ruin your carefully manicured indoor garden. Bug Eradicator kills over 1000 species of invasive pests that threaten the well being of your apartment farm. Keep out of reach of children; people of child-bearing age should avoid this product; may be harmful to the elderly. Lavender scented.
inventory	in stock
vendorID	2

Figure 23-20: *Here's an example of Dreamweaver's Master Detail Page Set. The screen on the left represents a master page—a list of items retrieved from a recordset. Clicking a link on this page opens a detail page (right), which displays the details of a single record.*

The Master Detail Page Set object automates the process of creating dynamic tables and recordset navigation bars, as well as adding many different server behaviors to your pages. In fact, there's nothing this tool does that you can't do (albeit more slowly) with the other tools you've learned about in this chapter.

Note: The Master Detail Page Set object isn't available for the ASP.NET server model.

To create a Master Detail Page set, follow these steps:

1. **Create two Web pages—one for the master page and another for the detail page.**

   They can be new, blank pages or existing pages that require the dynamic information from the database. It helps to use descriptive names for these pages, such as *productIndex.php* and *productDetails.php*. (Save each page with an extension appropriate for the server model you're using: .asp, .aspx, .cfm, .php, or .jsp.)

2. **Open the master page—the one listing all the records—and add a recordset to it.**

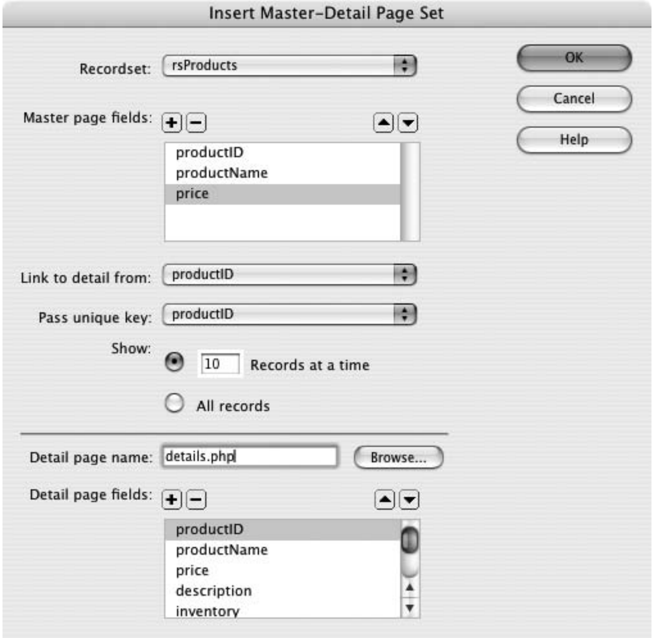
   This recordset must include not only all the columns to be displayed on the master page, but also all the columns to appear on the detail page. Both pages use the same recordset, so this one recordset must retrieve *all the* information you want on both pages. Also, make sure you select the primary key for the table (the ID used to identify each record). Even if you won't display this key on the page, you need it to link the master page to a detail page.

3. **In the Data tab of the Insert bar, click the Master Detail Page Set button (see Figure 23-1).**

   You can also choose Insert → Data Objects → Master Detail Page Set. Either method opens the Insert Master-Detail Page Set window (see Figure 23-21).

4. **From the Recordset pop-up menu, chose the name of the recordset you created in step 2.**

5. **Select the fields (database columns) you wish to appear on the master page.**



Insert Master-Detail Page Set

Recordset: rsProducts

Master page fields:
productID
productName
price

Link to detail from: productID

Pass unique key: productID

Show: ● 10 Records at a time
○ All records

Detail page name: details.php

Detail page fields:
productID
productName
price
description
inventory

Figure 23-21:
The Insert Master-Detail Page Set window lets you quickly create two common types of dynamic pages: one that lists many records in a database, and another that shows detailed information about a single database record. Unfortunately, this tool is not available in the ASP.NET server model in Dreamweaver.

You'll probably remove a bunch of columns from the "Master page fields" box, since most of the information is reserved for the detail page. To remove a column, click its name to select it and then click the minus (–) button. (If you accidentally delete a column, click the + button to add it back.) You can also change the order of the columns by selecting a column name and then clicking the up or down arrow buttons. The order the fields are listed in the box dictates the order in which they appear on the master page.

6. **From the "Link to detail from" pop-up menu, choose a column.**

   Here you're determining which item on the master page is linked to the detail page. In Figure 23-20, each product's name has a link to the detail page. If you're creating a staff directory, you might select the column that contains each staff member's name. In this way, visitors can click a name to see a page with that staff person's details.

7. **Using the "Pass unique key" field, select a column.**

   Choose a column that uniquely identifies a single record in the database, such as a product identification number or Social Security number. In most cases, this is the *primary key* in a database table (see page 769 for more on primary keys). You must add the "unique key" column to the recordset you create in step 2 for it to appear in this menu.

8. **Select how many records you wish to show.**

   You can either type a number into the "Records at a time" box or select "All records."

9. **Click Browse. Select the detail page you created in step 1.**

   In this step and the next, you're defining the detail page and what information appears on it.

10. **Select the fields you wish to display on the detail page.**

    The process is the same as step 5, but in this case, you'll probably include most, if not all, of the columns. After all, the detail page is where most of the information for an individual record shows up.

11. **Click OK to close the window and create the pages.**

    It may take a few moments for Dreamweaver to complete this step. It's adding a lot of code to both the master and detail pages.

Once completed, you can (and should) modify the tables, format the dynamic items, and design the page to your liking. Because the Master Detail Page Set tool just automates the process of adding repeating regions, recordset navigation bars, and other server behaviors, you must edit those items individually on the page. In other words, you can't return to the Insert Master-Detail Page Set window (Figure 23-21) to alter items on either page. For example, if you decide to remove a piece of dynamic information from the detail page, you must make this change on the detail page itself.

While the Master Detail Page Set tool makes building these types of pages a snap, it does have its drawbacks. The primary problem is that Dreamweaver uses the same recordset for both the master and detail pages. This can slow down the works, because even though you may want to display only a few columns of data on the master page (the productID, productName, and price fields in Figure 23-20), the recordset added to the page must retrieve *all* of the information the detail page needs. The database server is doing a lot of extra work retrieving unused data. However, there is a workaround: after creating the master and details pages, return to the *master page*, edit the recordset, and then select *only* the fields that that page uses. (Editing recordsets is described on page 799.)

Although the master-detail page set makes quick work of creating these types of pages, you can do all the same tasks using the tools you've already learned—repeating regions, recordset navigation bars, and so on—with the added benefit of greater design flexibility. For an example of creating a more complex master and detail page set by hand, complete the tutorial at the end of this chapter.

Passing Information Between Pages

Every now and then, you'll want to pass a piece of information from one page to another. The master-detail page set described in the previous section uses this concept: a link on the master page not only points to the detail page, but also passes along a unique ID used to create a filtered recordset on the detail page. In other words, the master page lists a bunch of records from a database. Each record not only links to the detail page, but also passes along its unique ID. The link might be something like *productDetails.php?productID=7*.

The information after the ? in the URL is a *URL parameter*, which the detail page uses to build a recordset. In this example, the detail page would find only one record—the one whose *productID* is 7—and display its details on the page. The key to the success of the master-detail page set, then, is the ability to pass information to another page and then use that information to filter a recordset (see page 785 for details on filtering database records).

Dreamweaver can pass information to other dynamic pages using either of two tools: the Go To Detail Page and Go To Related Page server behaviors.

Note: These two tools are available only for the ASP and JSP server models. For the other server models, several developers have created extensions to fill the gap. The free PHP Missing Tools extension by Felix One (*www.felixone.it/extensions/freeextdetailen.asp?IDProdotto=FX_PHPMissing*) includes both the Go To Detail Page and Go To Related page server behaviors for PHP. In addition, Deng Jie has created Go To Detail page extensions for ColdFusion, and ASP.NET. You can find these at the Adobe Exchange (see page 742)–just search for "Go To Detail Page." (These extensions work identically to the ones that come with Dreamweaver, so the following instructions still work.)

Go To Detail Page

A detail page, logically enough, is intended to provide more details on a single database record. To retrieve details on only one record, the detail page must include a recordset that filters the records of a database table based on some unique identifier, usually a record's primary key. For example, if every ad in a database of advertisements had its own unique ad ID, to find information on just a single ad you simply search for the one record that matches a particular ID number—in other words, you create a recordset that filters the data based on that ID number.

The Go To Detail Page server behavior provides a way to pass information to a detail page in the URL of a link—for example, *productDetails.php?productID=4*. In this way, when the detail page loads, it can use this URL variable—productID, in this example—to filter the database to retrieve the requested record (see page 789 for tips on using a URL variable to filter recordsets).

In many cases, you use this server behavior in conjunction with a repeating region (page 805) on a page that lists multiple database records. In other words, you use this behavior on a master page. Then you link each record in the list to the same page—the detail page—but pass along a different, unique ID for each record.

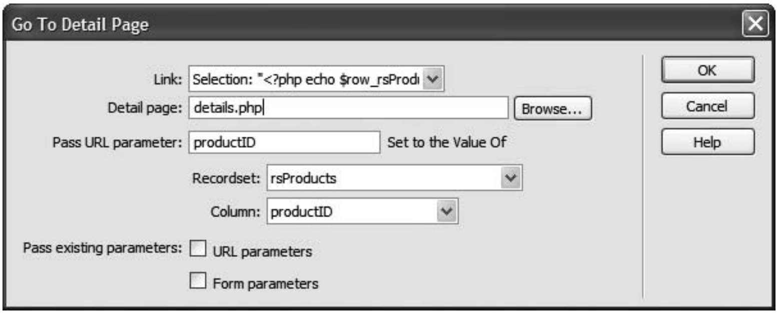
To use this server behavior, first add a recordset to the page. The recordset provides the information that's passed to the detail page (usually a record's primary key). Frequently, you'll also add a repeating region, or use Dreamweaver's Repeating Table command (page 803). Here's the whole process:

1. **Select an item on the page (graphic or text) to serve as a link to a detailed page.**

   Since you frequently use this server behavior in conjunction with a repeating region, you could select a piece of dynamic text within the repeating region. For example, on a page that lists many products, you could select the name of the product.

2. **Make sure the Server Behaviors panel is open (Ctrl+F9 [⌘-F9]). Click the + button; from the pop-up menu, choose Go To Detail Page.**

   The Go To Detail Page window appears (Figure 23-22). Skip the Link pop-up menu. (ASP and JSP users can also use either the Go To Detail Page button on the Data tab of the Insert bar or choose Insert → Data Objects → Go To → Detail Page to reach the Go To Detail Page window. If you've downloaded one of the extensions mentioned in the Note on page 818, you'll only find the Go To Detail Page option in the Server Behaviors panel.)



Go To Detail Page

Link: Selection: "<?php echo $row_rsProd

Detail page: details.php

Pass URL parameter: productID Set to the Value Of

Recordset: rsProducts

Column: productID

Pass existing parameters: □ URL parameters

□ Form parameters

Figure 23-22: *The Go To Detail Page server behavior can pass information from a recordset to another page by adding a URL parameter to a link.*

3. **Click Browse and select the detail page.**

   You're choosing a dynamic page, of course.

   Keep in mind that this server behavior doesn't actually *do* anything to the detail page, like adding a recordset to it. This behavior merely links to that page, passing along some additional information in the URL. In other words, to take advantage of the data passed along in the link, you need to add a recordset to the detail page that uses the URL variable. (You can also use the URL variable in other ways, as described on page 899.)

4. **Type a name into the "Pass URL parameter" box.**

   This is the name of the URL variable: For example, it's the word *productID* that appears in the URL *detail.php?productID=8.* Often, your best bet is to use the same name as the database column you select in step 6 below. But at the very least, the name should contain only letters and numbers, without spaces or other punctuation (because it's transmitted in the URL).

5. **From the Recordset pop-up menu, choose a recordset.**

   This is the recordset containing the information to be included in the URL.

6. **From the Column pop-up menu, select a database column.**

   This should be a primary-key field. This will be the *value* of the URL variable. In the example URL, *detail.php?productID=8,* it's the number 8. The actual number, of course, will be based on the primary key value for the particular record.

7. **If you want to pass on URL variables and form variables already present on the page, turn on the "URL parameters" and "Forms parameters" boxes.**

   For a discussion of these options, read the following section.

8. **Click OK to close the window and add the link.**

   Dreamweaver adds a link with all the programming that your site's server model needs.

Once you've completed these steps, open (or create) the detail page and add a recordset that uses the URL variable.

Go To Related Page

The other Dreamweaver tool for passing information between pages is the Go To Related Page server behavior. It takes information that's been passed to a page (in the form of URL parameters or Form parameters) and passes it onto yet another page.

Here's how you might use this feature. Suppose your Web site sells expensive manufacturing equipment. On a detail page, which lists the specifications for a particular machine, you include a link to a page with a larger photo of the machine. When the visitor clicks the "click for larger photo" link, the photo page retrieves and displays the proper photo.

But how does the photo page know which photo to display? An easy way would be to pass along the ID for the product, information that's included in the URL on the detail page.

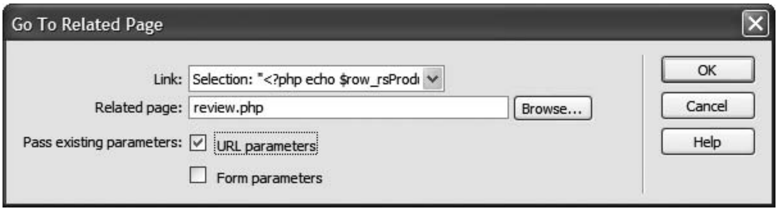
The Go To Related Page server behavior is ideal for this situation. It simply links to another page, passing along any URL parameters or form information already processed by the page. To use it:

1. **Select the graphic or text that will serve as the link to another page.**

   This could be text like "Click for larger photo," or "See a related review." Make sure the Server Behaviors panel is open (Ctrl+F9 [⌘-F9]).

2. **Click the panel's + button; from the pop-up menu, choose Go To Related Page.**

   The Go To Related Page window appears (Figure 23-23). Ignore the Link pop-up menu.



Go To Related Page

Link: Selection: "<?php echo $row_rsProd

Related page: review.php

Pass existing parameters: ☑ URL parameters

□ Form parameters

Figure 23-23: *This server behavior links from one page to another and passes along information contained in the URL, or data that was already sent to the first page as a form.*

3. **Click Browse. Select the page to which you want to pass the URL or form variables.**

   In order for the linked page to take advantage of these variables, you need to add a recordset and use the variables to filter the results. You can also add these variables to your Bindings panel, as described on page 897, and use them as dynamic text elements on the page, or even bind them to form elements.

4. **Turn on the appropriate boxes.**

   The "URL parameters" box passes along any URL variables to the linked page. For example, suppose you add this behavior to a Details page, creating a link to a page named *moreInfo.php*. When someone comes to the detail page, the URL might be *productDetails.php?productID=9* (depending on which record they're viewing). When the visitor clicks the link, the URL becomes *moreInfo.php?productID=9*. In other words, the *productID=9* from the detail page is sent along to the *moreInfo.php* page.

   The "Form parameters" box does the same thing for form variables. For example, when someone fills out and submits a Search form, the information travels to a Results page, which displays the results of that search. You could then preserve the form information submitted to the results page and pass it along to another page.

5. **Click OK to close the window and apply the server behavior.**

   To edit one of these behaviors, double-click its name in the Server Behaviors panel; to delete one, select it in the panel and press the – button or Delete.

Tutorial: Displaying Database Info

In this tutorial, you'll continue the work you started in the last chapter. Displaying the products available from CosmoFarmer's online store requires two dynamic pages. The first displays a list of all products available on the site. From that page, visitors can jump to a detailed description of an item for sale by clicking its name. You'll learn how to create both basic and advanced recordsets, and take advantage of some of Dreamweaver's built-in application objects.

This tutorial assumes you've done all of the setup work described in Chapter 22. If you haven't, turn to page 754 and follow the instructions for preparing the application server, database, and Dreamweaver for this project. (Also make sure you have XAMPP or MAMP up and running, as described on pages 757 and 759.)

Creating a Recordset

You'll start by opening an existing page and adding a recordset to it:

1. **Open *index.php* in the root folder of the local site you defined in the previous chapter.**

   Either choose File → Open and navigate to and select *index*, or double-click the file name in the Site Files panel.

The basic structure of this page is already complete. It was built using CSS layout, the Spry navigation bar, a table, Cascading Style Sheets, and the other HTML features you've already learned. There's nothing dynamic about this page yet, so you'll need to create a recordset and insert database information into the page.

2. **Open the Bindings panel (Windows → Bindings or Ctrl+F10 [⌘-F10]).**

   The Bindings panel is your command and control center for retrieving and using information from a database. It lets you create new recordsets and add dynamic data to a page.

3. **Click the + button in the Bindings panel and then choose Recordset.**

   You can also use the Server Behaviors panel (click the + button on that panel and then select Recordset); click the Insert Recordset button on the Data tab of the Insert bar (see Figure 23-1); or choose Insert → Data Objects → Recordset. Choose whichever method feels easiest for you. In any case, the Recordset box should now be on the screen. (Make sure you're using the *simple* mode—you should see a button labeled Advanced—as shown in Figure 23-24.)

   Next, select the information you want to retrieve.

4. **Type *rsProducts* into the Name box.**

   Since Dreamweaver lets you connect to more than one database, you must now indicate *which* database connection you want it to use.

5. **From the Connections pop-up menu, select "connCosmo."**

   This is the name of the connection you created in the last chapter.

   The *CosmoFarmer* database contains several tables. For this page, you'll create an index of all products for sale. That information is in the Products table.

6. **From the Tables menu, choose "products." Click Selected.**

   You don't need to retrieve *all* the information from the Products table. Since this dynamic page will present a listing of all of the products, you need only basic information, like the name of the product, its price, and its inventory status. More details about each product will appear on a second page, to be created later in this tutorial.

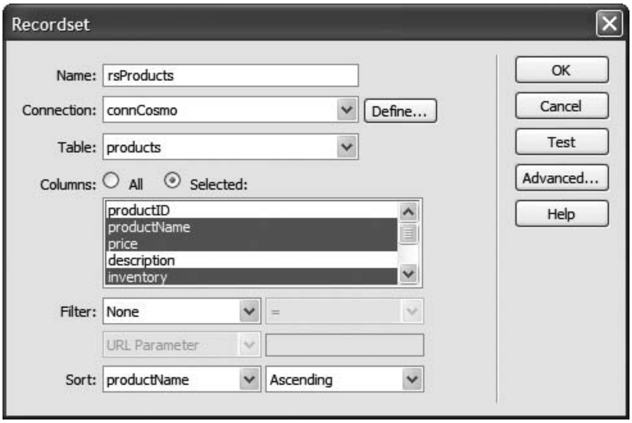
7. **In the Columns list, Ctrl-click (⌘-click) "productName," "price," and "inventory."**

   These are the columns of data you want to get from the database.

   You don't have to filter this recordset (meaning you're not trying to search the database for a particular product), so you can ignore these controls in the dialog box—but it would be nice if the product names appeared in alphabetical order.

8. **From the first Sort pop-up menu, choose "productName." From the second menu, choose Ascending.**

The Ascending option makes certain the records start with products whose names begin with A and end with names that begin with Z. The Recordset window should look like Figure 23-24.



Name: rsProducts

Connection: connCosmo

Table: products

Columns: ○ All ● Selected:

productID
productName
price
description
inventory

Filter: None =

URL Parameter

Sort: productName Ascending

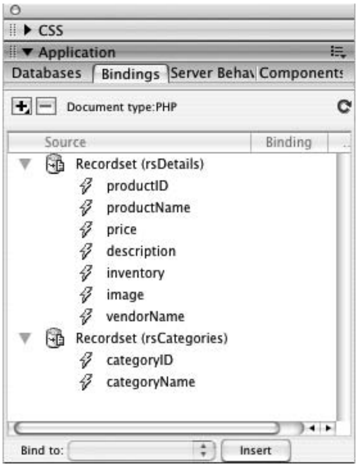
Figure 23-24:
Recordsets let you retrieve information from a database, so that you can display it on a page. This simple recordset will retrieve the product name, price, and inventory status of every product in the CosmoFarmer database.

9. **Click OK to close the Recordset dialog box.**

The new recordset appears in the Bindings panel. To see the database columns in the recordset, you need to expand the recordset list.

10. **In the Bindings panel, click the + icon (flippy triangle on Mac) next to the Recordset icon.**

The Bindings panel should look like Figure 23-25. There's already an HTML table on the page—below the CosmoFarmer Online Store headline—waiting for the product information.



Recordset (rsDetails)
productID
productName
price
description
inventory
image
vendorName
Recordset (rsCategories)
categoryID
categoryName

Figure 23-25:
The Bindings panel provides a list of a page's recordsets and data fields. You can use these fields to add dynamic text to a page.

11. **In the document window, click inside the table cell directly below the one labeled Product.**

    In this cell, you'll add the name of the item for sale.

12. **In the Bindings panel, click "productName" and then click Insert.**

    You've just added dynamic data to the page. For the moment, it looks like *{rsProducts.productName}*. You can apply formatting to dynamic data just as you would to regular HTML text—apply a style, change the font, make it bold, and so on.

13. **Make sure *{rsProducts.productName}* is selected in the table cell on the page, and then click the B button in the Property inspector.**

    Now you'll add both the price and inventory status to the page. There's an even easier way to insert recordset data.

14. **From the Bindings panel, drag "price" to the empty cell (on the Web page)—below the cell with the label "Price." Repeat this step by dragging "inventory" into the third empty cell ("Inventory Status") on the page.**

    At this point, your page should look like Figure 23-26.

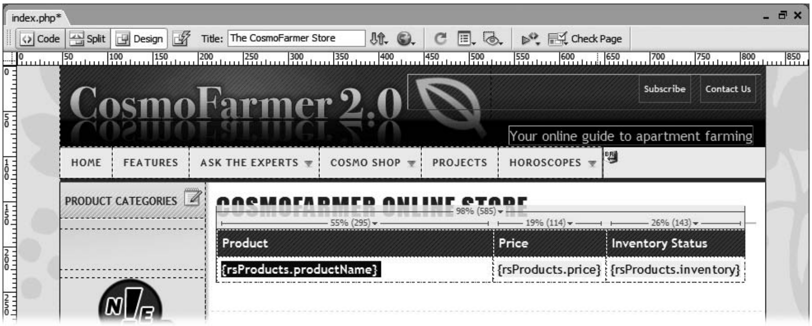


***Figure 23-26:** After you add dynamic information to a page, it appears with a blue background. Here, three pieces of dynamic data appear on the page from the rsProducts recordset: rsProducts.productName, rsProducts.price, and rsProducts.inventory. To change this background color, choose Edit → Preferences (Dreamweaver → Preferences). Click the Highlighting category and change the Live Data View colors at the bottom of the window.*

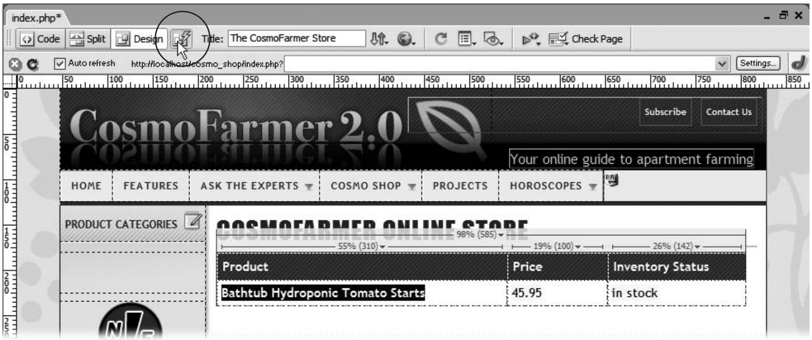
Live Data View and Creating Repeating Regions

When you add dynamic data to a page, it doesn't look like much. All you see is the recordset and column name between braces ({rsProducts.productName}, for example). Not only can this interfere with your design, it certainly doesn't give you a clear picture of what your Web page is actually going to look like.

Thank goodness for Dreamweaver's Live Data view:

1. **Click the Live Data View button in the document window's toolbar** (circled in Figure 23-27).

   Dreamweaver connects to the testing server and database to retrieve the data requested by the recordset (this may take a few seconds). For the first time, you get to see the page as it will appear on the Web. But there's a problem: only one item is listed. This page is meant to show listings for *all* products. To show more products, you have to add a repeating region—a part of the page that repeats for each record in a recordset.



Product	Price	Inventory Status
Bathtub Hydroponic Tomato Starts	45.95	in stock

***Figure 23-27:** When you click the Live Data View button (circled) in the document toolbar, Dreamweaver pulls data from the database and displays it in the document window. In addition, a second toolbar appears between the document and the document toolbar that lets you control various Live Data view settings.*

2. **Click the Live Data View button again to turn it off.**

   Dreamweaver runs a little more quickly when Live Data View is off, so it's generally a good idea to only turn it on when you really want to get an accurate view of the page in Design view.

3. **Move your cursor to the left of the phrase "Bathtub Hydroponics Tomato Starts" (you want your cursor to be over the left edge of the table); click when the right-pointing arrow appears.**

   That's how you select the bottom row of the table. (For other methods of selecting a table row, see page 252.) Since this row displays the info for a single product, it's a perfect candidate for a repeating region, where an additional row appears in the table for each product.

4. **On the Data tab of the Insert bar, click the Repeated Region button** (see Figure 23-1).

   The Repeat Region dialog box (Figure 23-13) appears, so that you can select which recordset to use (if the page has more than one) and how many records to display. In this case, since you have only one recordset, you just have to tell Dreamweaver how many records to show.

5. **In the "Records at a Time" box, type *10.* Click OK.**

   You don't know how many products the *CosmoFarmer* store offers at any time. If it were a lot, you wouldn't want to show them all on a single page—a thousand listings would make a pretty long Web page. In this case, just list ten records at a time.

   If the database has more than ten products, you need to provide a way for people to see the other items. You'll do that next.

6. **Click to the right of the table that lists the products. Then, in the Data tab of the Insert bar, click the Recordset Paging button and select Recordset Navigation Bar from the menu** (see **Figure 23-16**).

   The Recordset Navigation dialog box appears. There's only one thing to do here.

7. **Make sure the Text button is selected; click OK.**

   Dreamweaver plops a table containing four columns onto the page. The columns contain links that let visitors navigate the product listings.

   The table will look better with some CSS style applied to it.

8. **In the document window, select the table. From the class pop-up menu in the Property inspector, choose "paging".**

   For techniques on selecting a table, see page 252.

9. **Click at the end of the headline "CosmoFarmer Online Store" and then press Enter to create a new, empty paragraph; choose Recordset Navigation Status from the Record Count menu** (**see Figure 23-17**)**. Click OK when the Recordset Navigation Status window appears.**

   Dreamweaver inserts something that looks like this: *Records {rsProductsFirst Record} to {rsProductLastRecord} of {rsProductsTotalRecords}*. That's placeholder code for this notation: "Records 1 to 10 of 27." If you click the Live Data View button, that's what you'll see.

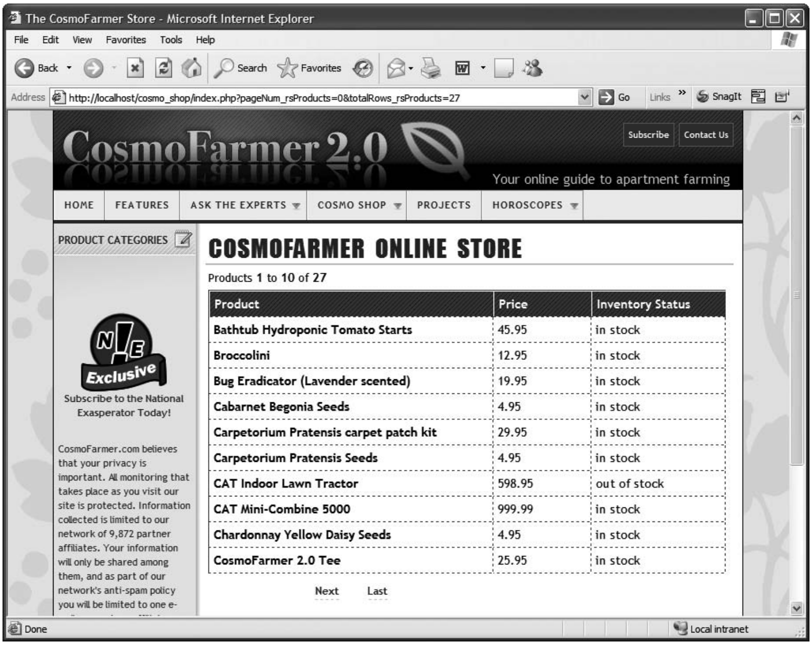
   Now to rework the message a little bit.

10. **Select the word Records and change it to Products; select each of the placeholders** (***{rsProducts.FirstRecord},*** **and so on**) **and then click the B button in the Property inspector to make them bold.**

    You can add or remove any of the non-dynamic text in the navigation status bar. Now to see the results of your hard work.

11. **Press F12** (**Ctrl-F12**) **to preview the page in your Web browser.**

    You may get a warning message titled "Update Copy on Testing Server." Even though when you first defined the site you told Dreamweaver that you're running the testing server on your own computer, it somehow thinks that the testing server is on another computer. Turn on the "Don't show me this message again" checkbox and click OK. The page opens in a Web browser displaying ten records (see Figure 23-28).


The CosmoFarmer Store - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/cosmo_shop/index.php?pageNum_rsProducts=0&totalRows_rsProducts=27
Go Links SnagIt
CosmoFarmer 2.0
Subscribe
Contact Us
Your online guide to apartment farming
HOME
FEATURES
ASK THE EXPERTS
COSMO SHOP
PROJECTS
HOROSCOPES
PRODUCT CATEGORIES
NE
Exclusive
Subscribe to the National Exasperator Today!
Next
Last
Done
Local intranet

CosmoFarmer.com believes that your privacy is important. All monitoring that takes place as you visit our site is protected. Information collected is limited to our network of 9,872 partner affiliates. Your information will only be shared among them, and as part of our network's anti-spam policy you will be limited to one e-

COSMOFARMER ONLINE STORE

Products 1 to 10 of 27

Product	Price	Inventory Status
Bathtub Hydroponic Tomato Starts	45.95	in stock
Broccolini	12.95	in stock
Bug Eradicator (Lavender scented)	19.95	in stock
Cabarnet Begonia Seeds	4.95	in stock
Carpetorium Pratensis carpet patch kit	29.95	in stock
Carpetorium Pratensis Seeds	4.95	in stock
CAT Indoor Lawn Tractor	598.95	out of stock
CAT Mini-Combine 5000	999.99	in stock
Chardonnay Yellow Daisy Seeds	4.95	in stock
CosmoFarmer 2.0 Tee	25.95	in stock

Figure 23-28: *Building dynamic Web pages in Dreamweaver is easy. In just a few short steps—all right, 25 steps—you can create pages for viewing database records. (You can click the Next link at the bottom of the page to jump to the next set of product listings.)*

Editing a Recordset and Linking to a Detail Page

Now that the main products listings page is complete, you need to create a link to the name of each product that, when clicked, opens a page with the details for that item. How does the detail page know which product to display? The link for each product will have some additional information—the product's ID number—tacked on to the end. In other words, while all of the product links will point to the same page, they'll pass some additional information which will help the detail page know which product to display.

Unfortunately, Dreamweaver doesn't provide a simple, one-click method to add a URL parameter to tell a detail page which recordset's details to display. You'll have to create that yourself, but first you need to add a primary key to the product's recordset.

1. **Open the Server Behaviors panel by pressing Ctrl+F9 (⌘-F9), or clicking the Server Behaviors tab in the Application panel group.**

   A list of all the different server behaviors appears; these were added when you created a recordset, put dynamic text on the page, and used Dreamweaver's other dynamic-page-creation tools. (Server behaviors are discussed in depth in Chapter 25.)

   Instead of adding another server behavior at this point, you can edit one you've already created: the recordset. When you first added this recordset, an important piece of information was missing—the product's ID number. (Actually, it was omitted from the tutorial intentionally, so that you'd now have this engaging educational opportunity to learn how to edit a recordset.)

Each product has its own ID number, which you'll use to tell the Details page which item to display.

2. **Double-click "Recordset (rsProducts)" in the Server Behaviors panel to open the Recordset dialog box.**

   You'll just add one additional column to the recordset.

3. **Ctrl-click (⌘-click) "productID" in the columns list, and then click OK.**

   You've just added one additional column (productID) to the recordset. Now the recordset not only retrieves the name, price, and inventory status for each product, but also its unique ID number.

4. **In the document window, select the dynamic data containing the product's name: "{rsProducts.prodName}."**

   A simple link to an already created Web page doesn't work here. Since the page containing a product's details is dynamic—it changes based on which product is being viewed—you need one of Dreamweaver's server behaviors.

   Unfortunately, Dreamweaver's PHP Server Model doesn't include the handy Go To Detail Page server behavior that you can find in some of the other server models. This means you'll have to create the link the long way. (There's a very handy extension named "PHP Missing Tools" which adds the Go To Detail Page server behavior for PHP. See page 818 for the details.)

5. **In the Property inspector, click the "Browse for file" icon.**

   The Select File window appears. This is the same process as creating a regular link.

6. **Locate and select the file *product.php,* located in the site's root folder.**

   This is the page that will display the detailed records of each product. Next, you'll add a little dynamic information to the link, so that each product's unique ID number can be passed along to the details page.

7. **Click the Parameters button on the right side of the Select File window (it's just to the right of the URL box).**

   The Parameters window opens (see Figure 23-29).

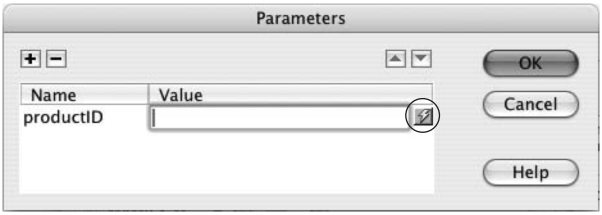


Figure 23-29: *The Parameters window lets you tack on dynamic information to the end of a link. In this way, you can pass information (for example, a product's ID number) off to another dynamic page that uses that information to display dynamic information.*

8. **Type *productID* in the Name column and then click the dynamic value button under the Value column (circled in Figure 23-29).**

   The Dynamic Data window appears (see Figure 23-30).

   ProductID is the name that will be added to the link. It will help the details page identify the purpose of the value (which you'll select next).

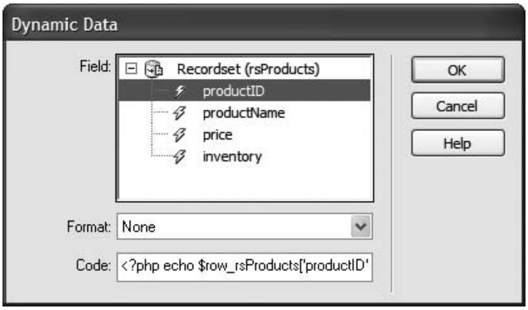


Figure 23-30:
The Dynamic Data window lets you select data from a database recordset to be used as part of the URL of a link.

9. **Expand the Recordset list (click the plus or arrow buttons to the left of the Recordset); select productID, and then click OK to close the Dynamic Data window.**

   This step selects the productID column from the recordset. In other words, this indicates that each link should have the unique ID for a particular product.

10. **Click OK to close the Parameters window, and finally click the OK (Windows) or Choose (Mac) button in the Select File window to finish creating the link.**

    You've just created a dynamic link. Now, let's preview it.

11. **Press the F12 (Ctrl-F12) key to preview the page in your Web browser. Click the Bathtub Hydroponic Tomato Starts link.**

    The detail page loads…without any details! You'll get to that step in a moment. In the meantime, look at the URL in your Web browser's address bar. It should look something like this: *http://localhost/cosmo_shop/product.php?productID=1.* Notice the *?productID=1* tagged onto the page *product.php.* That's the information the details page needs in order to retrieve the proper product information. The two pieces of information—*productID* and *1*—are what's called a *key/value pair.* The key *(productID)* tells the details page which field to look in, while the value *(1)* identifies a particular product with an ID of 1. Hit your browser's back button and click a different product's link and you'll see that the ID number is now different.

Note: Sometimes when you preview a Web page from Dreamweaver, the page won't display the changes you just made. The browser has cached (see page 607) the old file, and isn't displaying your recent changes. Just press your browser's reload button when this happens. Internet Explorer is particularly prone to this problem.

12. **Save and close this page.**

Building the Detailed Product Page

In this part of the tutorial, you'll build a detail page that displays all the details for a particular product. In addition, you'll create an advanced recordset that combines information from two separate database tables:

1. **Open the file called *product.php* in the root folder of the site.**

   Either choose File → Open and navigate to and select *product.php,* or double-click the file name in the Files panel.

   Since this page displays the details for a product, it must retrieve data from the database. To set this up, start by creating a recordset.

2. **Click the Insert Recordset button on the Data tab of the Insert bar (see Figure 23-1).**

   You can also choose Insert → Data Objects → Recordset. Or click the + button on either the Bindings panel or Server Behaviors panel, and then select *recordset* from the pop-up menu. In any case, the Recordset dialog box should now be open.

   The *CosmoFarmer* database contains several database tables—one for product information, one for vendor information, one that lists the different product categories, and one that contains user information (user names and passwords). The details page will list not only information from the Products table (such as the product's name and price) but also the name of the product's vendor. Unfortunately, the Products table only contains the vendor's ID number, not its name, so the recordset must incorporate information from both tables. Because the basic panel of the Recordset dialog box doesn't let you retrieve information from more than a single database table, you have to use the advanced setting.

3. **Click the Advanced button on the right side of the Recordset window.**

   The Advanced Recordset dialog box appears.

   Unfortunately, this isn't one of the areas in which Dreamweaver is particularly user-friendly. It helps to understand SQL (see page 793)—or you can just take the following steps.

4. **In the Name field, type *rsDetails.* From the Connection menu, choose "connCosmos."**

   In the next few steps, you'll create an SQL query—essentially a line of programming code that asks the database for particular information that matches specific criteria. In this case, it's the information for a particular product.

5. **Click the + icon (flippy triangle) next to the word Tables in the Database Items list (at the bottom of the dialog box).**

   It expands to reveal the four tables of the database: Category, Products, Users, and Vendors.

6. **Click the + icon (flippy triangle) to expand the Products table.**

Your job is to select the information from this table that you want to display on the page.

7. **Select "productID"; click the Select button to the right.**

   Notice that Dreamweaver writes *SELECT products.productID FROM products* in the SQL box. This is SQL code for selecting a particular column of data from a table. You can now choose the other pieces of information.

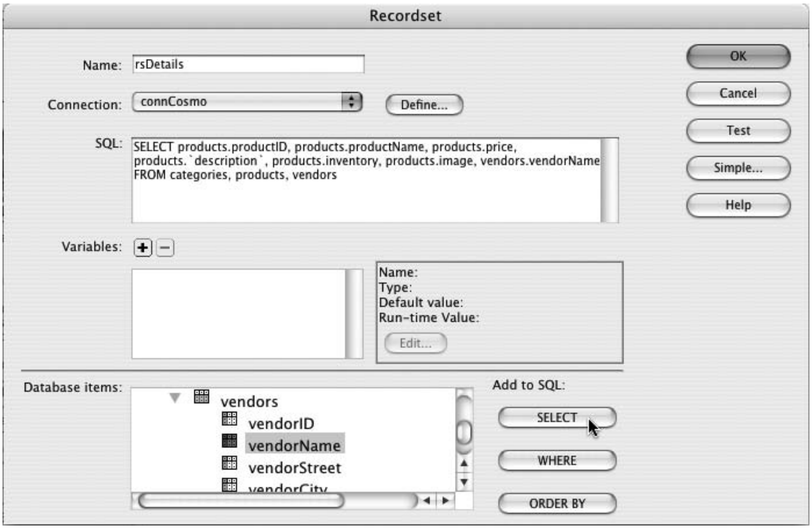
8. **Repeat step 7 for the following items in the Products table: productName, price, inventory, description, image.**

   These are all the items you need to retrieve from the Products table. Now you can choose data from the Vendors table.

Tip: If you understand SQL, you can bypass this point-and-click approach and simply write a SQL query directly in the SQL box.

9. **Click the + icon (flippy triangle) to expand the Vendors table. Repeat step 7 for the vendorName item in that table.**

   The dialog box should now look like Figure 23-31. Congratulations, you've just created a SQL query.



Name: rsDetails

Connection: connCosmo

SQL:

```
SELECT products.productID, products.productName, products.price,
products.`description`, products.inventory, products.image, vendors.vendorName
FROM categories, products, vendors
```

Figure 23-31: *The Recordset window's advanced mode lets you use a data tree (the bottom half of the window) to build an SQL statement. By selecting a column name and clicking either the SELECT, WHERE, or ORDER BY buttons, you can get Dreamweaver to do some of the heavy lifting. Unfortunately, you still need to understand a little bit of SQL to create functioning database queries using these advanced options. See page 793 for a brief introduction to SQL.*

But there's a problem here: Because of this query's structure, it retrieves *all* the records for both tables (click the Test button to see for yourself). To remedy the situation, you must do two things: first, combine information from two tables so that you get the vendor information for the corresponding product; and second, retrieve only the information for the particular product specified by the link you created in the last part of this tutorial.

10. **Click inside the SQL box after the word "vendors." Press Enter or Return.**

    You'll have to dive into typing out SQL code.

11. **Type *WHERE products.vendorID = vendors.vendorID*.**

    This little bit of code is called a *WHERE clause*, and it helps filter information in a database. In this instance, you've created what's called a *join*—a statement that joins two or more tables together. When you retrieve product information, you also want to retrieve the name of the vendor who manufactures that product. By matching the vendor ID from the Products table to the identical vendor ID in the Vendors table, the database can produce the proper vendor name.

    If your eyes are glazing over, go get a cup of coffee before plunging ahead.

12. **Click the + button to the right of the word Variables.**

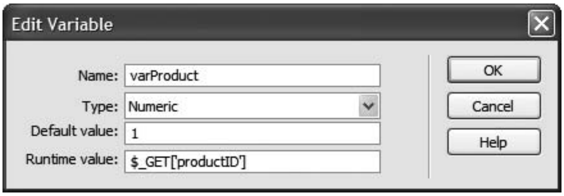
    The Edit Variable window appears (see Figure 23-32).

    You're about to expand on the WHERE clause you just wrote. Not only do you need to get the details of a product (plus the vendor's name), you also want to retrieve just a single record—the particular product whose details the visitor wants to review.

13. **Click in the Name box and type *varProduct*. Type *1* for "Default value." And, type *$_GET['productID']* for the "Runtime value."**

    The Edit Variable box should now look like Figure 23-32. Look back to step 8 in "Editing a Recordset and Linking to a Detail Page" (page 829). Remember that the ID number for the product is embedded in the URL that links to this page. In other words, when someone clicks a link on the main product listings page, the ID number for the product is passed along like this: *product.php?productID=12*.

    In this step, you're retrieving that information from the URL—that's the *$_ GET ['productID']* part—and storing it in a variable that you'll use in the rest of the SQL query.



Edit Variable

Name: varProduct

Type: Numeric

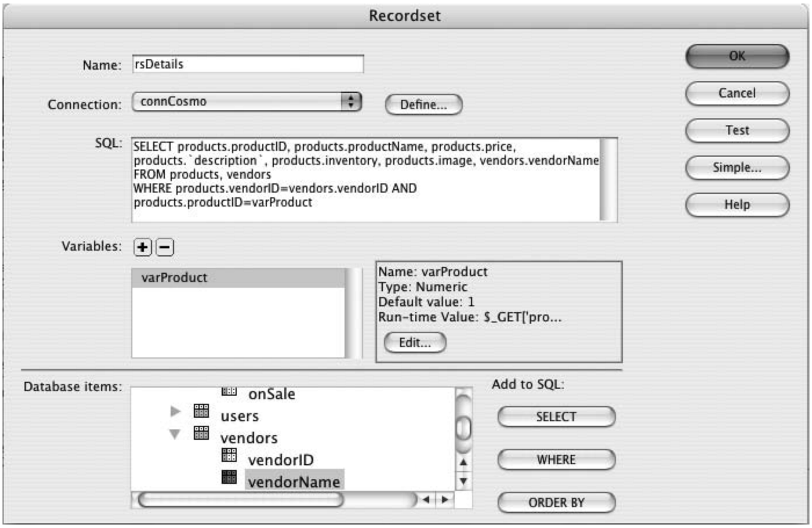
Default value: 1

Runtime value: $_GET['productID']

Figure 23-32:
When you add a variable to an SQL query you're letting that query respond to information from another source. In this example, information that's sent with the URL that's used to link to this page provides the unique ID number for each product. Because that URL will "vary" ("variable", get it?) the recordset will produce different results–the specific details for one product.

14. **Click OK to add the new variable. In the SQL box, click at the end of the WHERE clause (after vendors.vendorID) and type *AND products.productID = varProduct*.**

    The Recordset window should now look like Figure 23-33.



Name: rsDetails

Connection: connCosmo

SQL:

```
SELECT products.productID, products.productName, products.price,
products.`description`, products.inventory, products.image, vendors.vendorName
FROM products, vendors
WHERE products.vendorID=vendors.vendorID AND
products.productID=varProduct
```

Variables: varProduct

Name: varProduct
Type: Numeric
Default value: 1
Run-time Value: $_GET['pro...

***Figure 23-33:** In the Recordset window's advanced mode, you can create highly detailed SQL queries that can retrieve and merge information from multiple database tables.*

15. **Click the Test button to see if the SQL query works.**

    A Test SQL Statement window opens, containing a single record. Hallelujah: It includes not only product details but also the vendor's name. (If Dreamweaver spits out an error message instead of a record, there's probably a typo somewhere in the SQL. Either try to identify the problem, or delete the SQL query and start again at step 5 above.

16. **Click OK to close the window. Choose File → Save to save your changes.**

Filling in the Details

Now all you need to do is add the information retrieved in the recordset to the page:

1. **In the document window, select the words "Product name."**

   You'll add the name of the product here.

2. **Open the Bindings panel.**

   Either press Ctrl+F10 or click the Bindings tab in the Application panel group.

3. **In the Bindings panel, click the + icon (flippy triangle) next to the recordset to display all the columns retrieved in the recordset. Select "productName," and then click Insert.**

   The placeholder for the dynamic data—*{rsDetails.productName}*—appears in the document window. Next, you'll try the drag-and-drop method of inserting dynamic data.

4. **In the Bindings panel, drag "description" into the empty, blank line just below the product name.**

   Now it's just a matter of adding the additional data to the page.

5. **Continue adding content to this page using these same steps.**

   Add the price, vendor name, and inventory status in the appropriate places in the document window.

   To finish off this page, add a photo to the page.

6. **Click just to the left of the product description (just before {rsDetails.description}), Choose Insert → Image.**

   The Insert Image window appears. You've encountered this dialog box many times before when you inserted a graphic (see Chapter 6). However, in this case, you'll retrieve the image's file name from the database.

7. **Select the Data Sources radio button.**

   In Windows, this button is at the top of the Insert Image window; on a Mac it's at the bottom. At this point, depending on whether you're using Windows or a Mac, you'll either see the Select Image Source window or the Dynamic Data window (see Figure 23-34).

   A list of all of the different data items from the recordset appears. See the item labeled "image"? The database stores the *name* of the image file for each product in the database. But the image file itself isn't stored there. Although some databases do let you store the actual image data—called binary data—in a database, if you merely want to display an image on a Web page, that's a big waste of database space and processing power. A better method, and the one used here, is to just store the name of an image file that's already in the Web site (for example, in a folder named *images*). In the next two steps, you'll craft a path to an already existing file on the Web server.

8. **Select "image".**

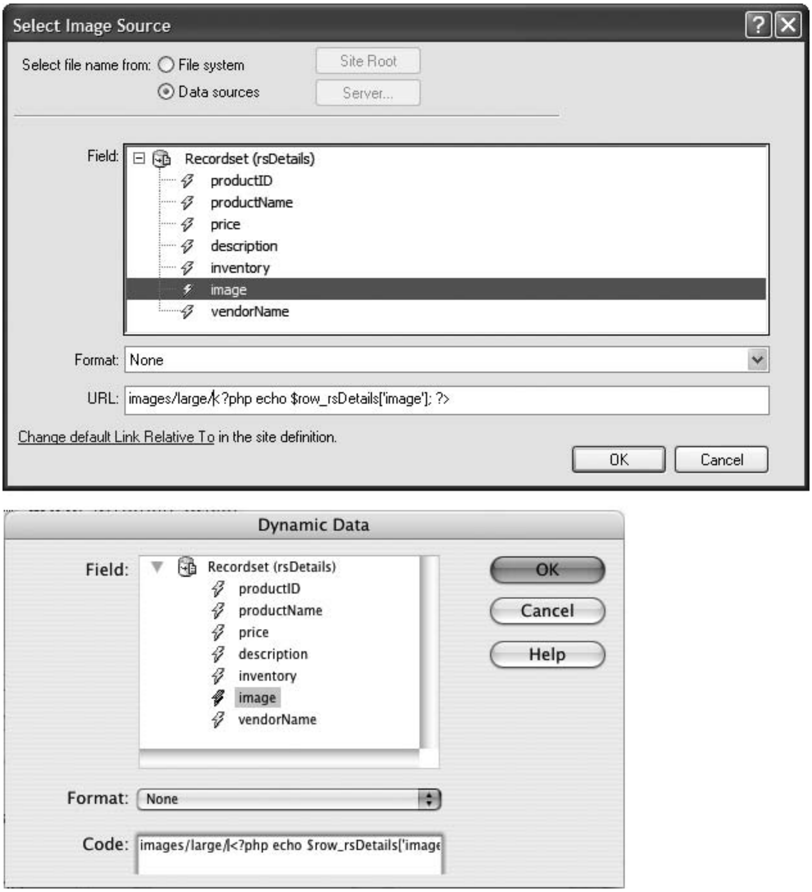
   At the bottom of the window (in the URL box [Windows] and the Code box [Mac]), you'll see some PHP programming code that looks like this: <?php echo $row_rsDetails['image']; ?>. It looks scary, but it simply prints whatever is stored in the image column for the particular recordset. Since the image column in this table stores the name of a file, this code would print something like *tomatoes.jpg*. This is the value that will appear as the src property of the image—in other words, specifying where a Web browser should look for the file. (You learned about the src property in Chapter 6.)

Tip: One benefit of this only-store-the-file-name-in-the-database approach is that you can store different size images for the same record in different locations, yet still use the same recordset field for each. For example, say you had a dynamic page that displays thumbnails of all the products in your database. And you also have a page that displays a larger picture of the same product. Use the same file name in each case (*tomato.jpg* for example) but store the thumbnail file in one location (perhaps *images/small/*) and the larger file in another folder (*images/large/*). When you want to insert a thumbnail you could use the database field used to store the image, but add the appropriate path (for example, *images/small/*). When you want to insert the larger image you could use the same database field (*image,* for example) but supply the path that points to the larger file (*images/large/*).

For the CosmoFarmer store, the images are actually stored inside a folder named large, which is inside the images folder. (Thumbnails for each image are stored in the small folder.) So for the image to appear correctly, you need to add a little additional information to this window.

9. **In the URL (Code) box, click before the text** *<?php* **and type** *images/large/* **as pictured in Figure 23-34.**

   Don't omit the forward slash after the word large. You'll then have this code: *images/large/<?php echo $row_rsDetails['image']; ?>*.



Select file name from: ○ File system ● Data sources

Field: Recordset (rsDetails)
- productID
- productName
- price
- description
- inventory
- image
- vendorName

Format: None

URL: images/large/<?php echo $row_rsDetails['image']; ?>

Field: Recordset (rsDetails)
- productID
- productName
- price
- description
- inventory
- image
- vendorName

Format: None

Code: images/large/<?php echo $row_rsDetails['image

Figure 23-34: *Paths for links and images needn't be hardwired into a Web page. You can craft custom paths and links using information retrieved from a database. A good technique when working with images and a database is to store just the name of the image file in the database. Then, when you insert an image using a data source from a database, you select the field that contains the file name and then precede the PHP code that appears in the URL (Windows, top) or Code (Mac, bottom) box with the information necessary to create a complete path to the image.*

10. **Click OK. (If the Image Tag Accessibility Attributes window appears, just click OK to dismiss it.)**

    A little square icon appears in the document window. This is an image placeholder icon, and represents the space where an image will appear when the page is viewed in a Web browser. Next you'll add a style to this graphic.

11. **Make sure the little square icon is still selected (if not, click it), and then choose *productImage* from the Class menu.**

    The image placeholder floats to the right side of the page. One last thing to do: because you're a Web design expert (or at least an aspiring expert), you know that images should always have their *alt* property (page 209) set. But since the exact image that appears will vary based on which product is being viewed—it might be a tomato or a picture of an indoor lawn mower—how can you specify alt text that matches the picture? As with most things in this section of the book, the answer is, "You do it dynamically, of course!"

    To set alt text dynamically you need to use the Tag inspector window.

12. **Make sure the image placeholder is still selected, and then choose Window → Tag Inspector (or press F9).**

    The Tag inspector is a kind of super Property inspector. It lets you set every conceivable HTML property for a particular tag (not just the most common ones listed in the Property inspector).

13. **Click the "Show List View" button (the icon with A-Z on it), and click the empty area to the right of "alt".**

    You could just type the alt text here, but notice that our friend, the lightning bolt icon, appears. This is your clue that we can access some dynamic data.

14. **Click the lighting bolt icon to open the Dynamic Data window.**

    The Dynamic Data window works just like the Dynamic Text window pictured back in Figure 23-8. It let's you insert information from a field in a recordset. In this case, the appropriate text description for each product's photo is the product's name.

15. **Click *productName* in the Dynamic Data window; click OK.**

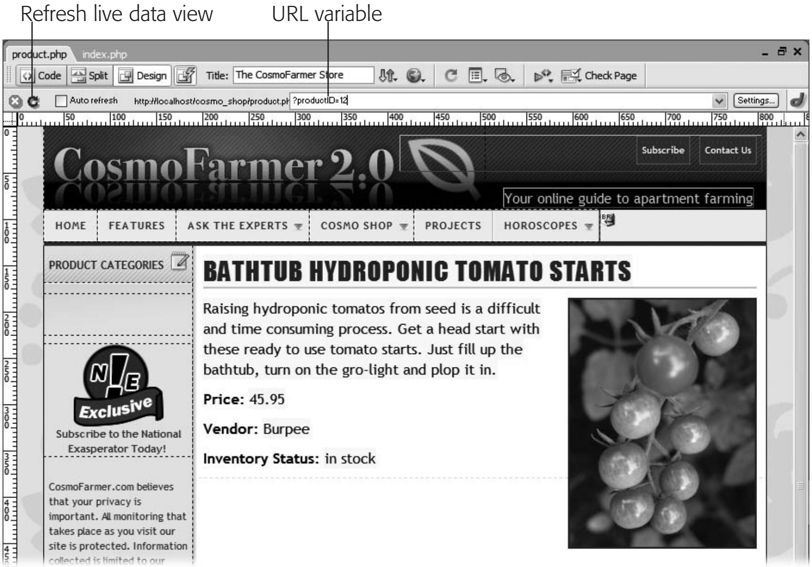
    Now whenever the page is viewed, the alt text for the image will match the product's name. Time to see how it looks.

16. **Click the Live Document View button. If everything looks good (Figure 23-35), choose File → Save.**

To see the results of your hard work, open the *products.php* page in Dreamweaver and then press the F12 key (option-F12) to preview the page in a browser. Now click a link to see the details for that product.

Operators Standing By

One final touch would make the products page perfect. Each product sold at the *CosmoFarmer* online store belongs to a category—plants, seeds, pest control, and so on. Shoppers might find it useful to view a list of products in a particular category they're interested in—just the plants, for instance.



Title: The CosmoFarmer Store

http://localhost/cosmo_shop/product.ph ?productID=12

BATHTUB HYDROPONIC TOMATO STARTS

Raising hydroponic tomatos from seed is a difficult and time consuming process. Get a head start with these ready to use tomato starts. Just fill up the bathtub, turn on the gro-light and plop it in.

Price: 45.95

Vendor: Burpee

Inventory Status: in stock

CosmoFarmer.com believes that your privacy is important. All monitoring that takes place as you visit our site is protected. Information collected is limited to our

Figure 23-35: *You can preview additional records by modifying the URL that appears just below the document toolbar. In the box just to the right of product.php?, type: productID=12. The information for the product with ID number 12 appears (you may need to press the Refresh button to see the new record). Try other numbers to view other products.*

Since category information is stored in the database, you can use this info to create such a feature. In this final part of the tutorial, you'll add a category navigation bar along the left side of the products page, so that when a visitor clicks a category name, a list of the products within that category appears:

1. **Open the *products.php* page.**

   This is the page to which you add the category links. The first step is to add a new recordset that retrieves all of the category names from the database.

2. **Open the Bindings panel (press Ctrl+F10, or click the Bindings tab in the Application panel group), click the big + button, and then select Recordset from the pop-up menu.**

   You can also click the Insert Recordset button on the Insert bar's Data tab (see Figure 23-1); choose Insert → Data Objects → Recordset; or click the + button on the Server Behaviors panel and then select Recordset from the pop-up menu. In either case, the Recordset dialog box opens. (You may need to click the Simple button to switch the Recordset dialog box out of advanced mode.)

3. **Type *rsCategories* into the Name box.**

   The *CosmoFarmer* database includes a table with the names of each category of products sold. You'll use this table to dynamically generate the list of category names.

4. **From the Connections pop-up menu, select "connCosmo."**

   This is the same connection you've used throughout this tutorial. In this case, you're going to retrieve information from a different table in that database.

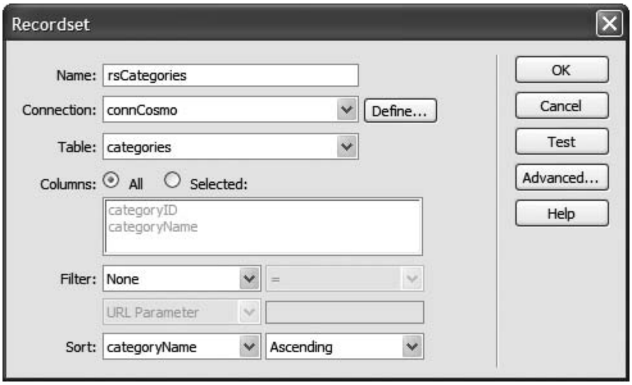
5. **Select Categories from the Table menu.**

   The Categories table is very basic: just a name and ID number. The Products table identifies a category by using a Category ID number—the table's categoryID field. You may wonder why a separate table is even necessary. Why not just store the category name with the product information?

   This design has two advantages. First, because the table is just a list of category names, you can easily retrieve an alphabetized list of those names by creating a recordset. That ability is useful, for example, when you want to add a list of categories to a page—as in these tutorial steps. In addition, the separate category table makes changes to categories easier. If you decide you want to change a name—say "Pest Control" to "Pest and Weed Control"—you need to update it only in one record in the Category table. If "Pest Control" were stored in the Products table, you would have to change the name to "Pest and Weed Control" in potentially hundreds of records.

6. **Make sure the All radio button is selected, and then choose "categoryName" from the Sort box.**

   At this point, the dialog box should look like Figure 23-36.



Name: rsCategories
Connection: connCosmo
Table: categories
Columns: ● All ○ Selected:
categoryID
categoryName
Filter: None =
URL Parameter
Sort: categoryName Ascending

Figure 23-36:
By storing all the product category names in a single table, you can build a dynamic category navigation bar with the help of a simple recordset.

7. **Click OK to apply the recordset to the page.**

   Now the page has two recordsets—one to retrieve product info, the other to retrieve the list of categories. You'll add the category name to the page next. The Bindings panel should look like Figure 23-37. The list of category links will appear in the left-hand sidebar on the page; currently the text "Product Categories" appears in that space. You'll add the links below that. However, to make the job easier, you'll first hide the style sheets that create the design for this page.

8. **Choose View → Style Rendering → Display Styles.**

   Doing so turns off the Display Styles option and temporarily hides the style sheets applied to this page. Why? Sometimes, the styles you create can make such a drastic change to the look of the page that it's hard to select, edit, and

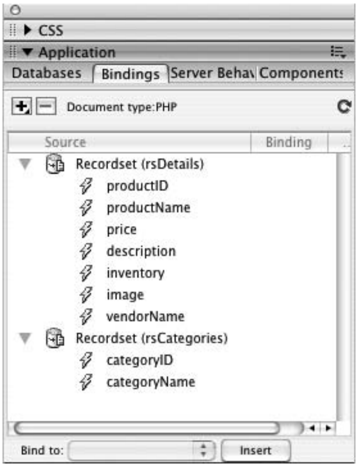


Figure 23-37:
The Bindings panel displays all recordsets currently applied to a page. To hide the recordset's field names, click the minus (–) sign (arrow on the Mac) to the left of the recordset icon.

insert text and other HTML in Dreamweaver's Design view. This left-hand sidebar is a good example. The list of category links will be a bulleted list of names, but the current styles hide the telltale sign of a list item (the actual bullet) and make it hard for you to see exactly where to place your cursor at the beginning of a list item. (Bulleted lists are discussed on page 92, and you can learn more about Dreamweaver's style rendering options on page 304.)

9. **Scroll down the page a bit, until you spot the empty bullet that appears below the "Product Categories" headline and directly above the National Exasperator logo (the "N!E" graphic).**

   That's an empty bulleted item—you'll place the category name there.

10. **From the Bindings panel, drag "categoryName" just to the right of the bullet.**

    This adds the dynamic text for the category name to the page. Next, you'll add a repeating region so that *all* the category names appear.

11. **In the Tag selector at the bottom of the document window click the <li> tag.**

    The Tag selector (page 22) is the most accurate way to select an HTML tag. When working with Dreamweaver's Repeat Region server behavior you have to be particularly careful that you've selected exactly what you want to repeat for each record in a recordset. In this case, you want each category name to appear as its own bulleted item in the list—that means the <li> tag (or list item) needs to be selected before applying the Repeat Region server behavior.

12. **On the Data tab of the Insert Bar click the Repeated Region button (see Figure 23-1).**

    The Repeat Region window appears.

13. **Select "rsCategories" from the menu. Click the All records radio button, and then click the OK button to create the repeating region.**

    If you preview the page at this point, you see a list of categories along the left side of the page. To make this list a functional navigation bar, you'll add a link to the category name.

14. **Select the dynamic text *{rsCategories.categoryName}* in the document window. In the Property inspector, click the "Browse for file" icon.**

    The Select File window appears. This is the same process as creating a regular link.

15. **Locate and select the file *category.php,* located in the site's root folder.**

    This is the page that will display the list of products within a particular category. You'll next add a little dynamic information to the link, so that each category's unique ID number can be passed along.

16. **Click the Parameters button on the right side of the Select File window (it's just to the right of the URL box).**

    The Parameters window opens.

17. **Type *categoryID* in the Name column and then click the Dynamic Value button (the lightning bolt) under the Value column.**

    The Dynamic Data window appears (see Figure 23-30).

    categoryId is the name that will be added to the link. It will help the details page identify the purpose of the value (which you'll select next).

18. **Expand the *rsCategories* recordset list (click the plus or arrow buttons to the left of the Recordset); select categoryID and then click OK to close the Dynamic Data window.**

    This step selects the *categoryID* column from the recordset. In other words, this indicates that each link should have the unique ID for a particular category.

19. **Click OK to close the Parameters window and, finally, click the OK (Windows) or Choose (Mac) button in the Select File window to finish creating the link.**

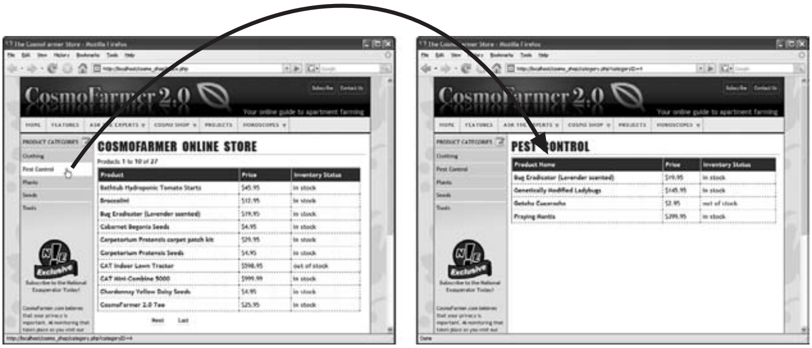
    You've just created a dynamic link. Since the hard work is now done, you can turn the style sheets back on.

20. **Choose View → Style Rendering → Display Styles.**

    The document window returns to all of its green and blue glory. Now, let's preview the page.

21. **Press F12 (Option-F12). When the page opens, click any of the category names at the left side of the page.**

    A new page should open, listing all of the products within a particular category, as shown in Figure 23-38.



COSMOFARMER ONLINE STORE

Products 1 to 10 of 27

Product	Price	Inventory Status
Bathtub Hydroponic Tomato Starts	$45.95	in stock
Broccolini	$12.95	in stock
Bug Eradicator (Lavender scented)	$19.95	in stock
Cabernet Begonia Seeds	$4.95	in stock
Carpetorium Pratensis carpet patch kit	$29.95	in stock
Carpetorium Pratensis Seeds	$4.95	in stock
CAT Indoor Lawn Tractor	$598.95	out of stock
CAT Mini-Combine 5000	$999.99	in stock
Chardonnay Yellow Daisy Seeds	$4.95	in stock
CosmoFarmer 2.0 Tee	$25.95	in stock

Next Last

PEST CONTROL

Product Name	Price	Inventory Status
Bug Eradicator (Lavender scented)	$19.95	in stock
Genetically Modified Ladybugs	$145.95	in stock
Gotcha Cucaracha	$2.95	out of stock
Praying Mantis	$299.95	in stock

Figure 23-38:
URL parameters aren't just used to link to a page with the details of a single record. You can use URL parameters to pass any information that can be used to search a database. In this case, the link on the products page (left) provides an ID number to the categories page (right), so that just the products within a particular category appear.

Congratulations! You've just built two powerful, complex, dynamic Web pages (and probably watched three presidential administrations pass). As you can see, Dreamweaver has an impressive array of tools for building dynamic pages. And even though there were some twists and turns to negotiate, you never once had to resort to the dreaded Code view.

CHAPTER

24

Web Pages that Manipulate Database Records

Just displaying database information on a Web page is useful, but you may be more interested in using the Web to *collect* information from your site's visitors (see Figure 24-1). Maybe something as simple as an online registration form will do the trick. Other times, you may have something more ambitious in mind—like a full-fledged e-commerce system that doesn't pay the bills unless it provides a way to collect product orders and credit card numbers.

Once you've got data in the database, clearly you need a way to update and delete that information. After all, prices change, products are discontinued, and you may suddenly want to remove any record of "Harvey the Wise Guy" from your site's online guestbook. Thankfully, Dreamweaver makes changing information in a database simple and painless.

Tip: You may feel more comfortable learning these concepts by *doing* them. If so, turn to the tutorial on page 867 before reading this next section.

Adding Data

As noted in Chapter 11, the primary method of collecting information over the Internet is the *HTML form*. Its basic elements—text boxes, radio buttons, pop-up menus, and so on—can collect a wide assortment of data. But to funnel this information into a database, you need to either write your own program or simply use Dreamweaver's built-in tools. With its Record Insertion Form wizard and Insert Record server behavior, Dreamweaver makes adding data a simple process.

Warning: You might not want just anyone adding, editing, or deleting database information. To control access to these types of pages–or any page, for that matter–use Dreamweaver's User Authentication server behaviors, as discussed on page 887.

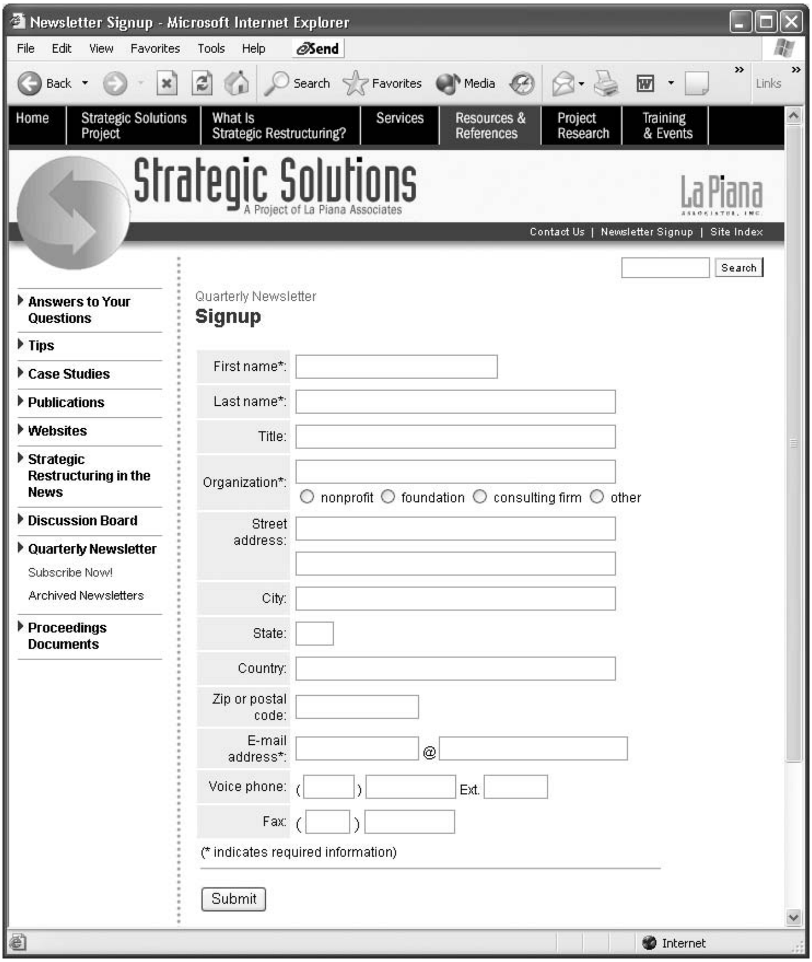


Figure 24-1: *Whether you're looking to accept credit cards on an e-commerce site or gather sign-up information for an online newsletter, Dreamweaver simplifies the process of creating forms that funnel info into a database.*

Dreamweaver's Record Insertion Form Wizard

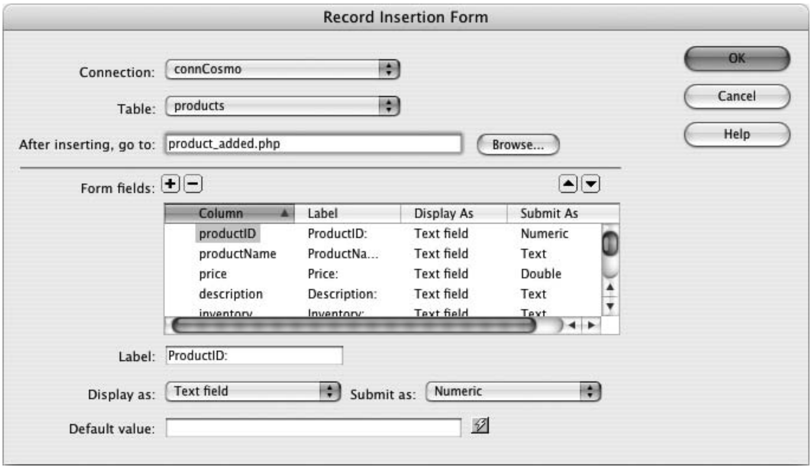
Dreamweaver's Record Insertion Form wizard is the quickest way to build a page for adding records to a database. It builds a form, creates a table, and then adds all the necessary programming code in just a couple of steps. To use it:

1. **Create a new Web page, and save it to your site (or open a dynamic Web page that you've already created).**

Make sure the page uses the extension (.asp, .aspx, .cfm, .php, .jsp) that matches the server model (page 751) of your site. (See page 766 for info on creating new dynamic pages.)

2. **In the document window, click where you want the form to appear. Choose Insert → Data Objects → Insert Record → Record Insertion Form Wizard.**

   You can also select Record Insertion Form Wizard from the Insert Record menu on the Insert bar's Data tab. Either way, the Record Insertion Form window opens (Figure 24-2).



Record Insertion Form

Connection: connCosmo

Table: products

After inserting, go to: product_added.php

Form fields:

Column	Label	Display As	Submit As
productID	ProductID:	Text field	Numeric
productName	ProductNa...	Text field	Text
price	Price:	Text field	Double
description	Description:	Text field	Text
inventory	Inventory:	Text field	Text

Label: ProductID:

Display as: Text field

Submit as: Numeric

Default value:

Figure 24-2: *When creating a form, it's a good idea to remove columns that the database creates for its own use, such as record ID numbers. Here, the column productID is a unique ID number created by the database. Whenever someone adds a new record, the database creates a new, unique product ID number. You wouldn't want anyone to tamper with this number, so leave it off the form.*

3. **From the Connection menu, select the database connection.**

   Dreamweaver needs a place to put the data your forms are collecting. If your site works with several databases, select the connection for the database to which this form will add information. (Database connections are described on page 776.)

4. **From the Table menu, choose a table.**

   A list of all tables available in the database appears in this menu. Dreamweaver lets you insert data only into a single table at a time. You can, however, add multiple record insertion forms to a single page. So it's possible to create one "add data" page that contains separate insert record forms for each table in the database. You would still have to input records one at a time for each table, but at least your visitors would have to visit only a single page in order to add a record to any table. (Of course, if your database has 20 tables, it would be a good idea to create several "add record" pages so you don't end up with one gigantic Web page containing 20 different forms.)

Tip: You can use Dreamweaver CS3's new Spry Tabs to display multiple insert forms on a single page without wasting a lot of precious screen real estate. The tutorial starting on page 867 shows how.

5. **Click the Browse button. Navigate to your Web site, and then select a file.**

   Choose the page your visitors will see after adding a record to the database. It could simply be a page saying, "Thanks for signing up with our Web site." Or if the insertion form adds a new employee to your company's employee database, you could choose the page that lists all the employees (including the one just added to the database). If you want people to be able to add multiple records, one right after the other, you might also select the record insertion form itself; that way, once visitors add one record, they'll return to the form again, ready to continue the very pleasant work of data entry.

Note for ASP.NET Users: On ASP.NET pages, you also have the option of specifying an "error page," which appears when the addition to the database fails for some reason.

   You can also change how the menu item for each database column is formatted on the page, as follows.

6. **In the "Form fields" box, select a database column, and change its settings, if you like.**

   Your options include:

   - **Remove the field.** Click the minus (-) button to remove the field from the form. It doesn't appear on the final form page, and users can't manually submit a value for this field. You should remove any fields that the database automatically fills out, such as a primary key field (see page 769). (If you accidentally delete a field, click the + button to add it back.)

Tip: Use the up and down arrow buttons (see Figure 24-2) to rearrange the order in which the fields appear in the form on the Web page.

   - **Label** is the text Dreamweaver adds next to the form field on the page. It identifies what someone should type in the field, like First Name. Dreamweaver just uses the name of the column in the database—fName, for instance—so it's usually best to change this text to something more understandable.

   - The **"Display as" menu** lets you select the *type* of form element you want to use to collect the column's information. If the column is someone's first name, select Text Field. This selection will add a text box, where visitors can type their names, to the form. On the other hand, if people are supposed to choose from a limited number of choices (*U.S. Postal Service, FedEx–2 day,* and *FedEx–next morning,* for example), you might select Radio Group instead.

     Radio buttons or pop-up menus can also ensure consistency and rule out typos. On a form that asks visitors to indicate the state they live in, you could offer a pop-up menu that lists the 50 states. If you provided a text box instead, your visitors would be able to type in every conceivable abbreviation and misspelling. (For a description of the different types of form elements, see page 404.)

Note: Dreamweaver can also create *dynamic* menus, which display data taken from a database. See page 860.

- The **"Submit as" menu** is automatically determined according to how you've set up your database. It tells Dreamweaver what kind of data the field contains: text, number, date, and so on. Dreamweaver figures this information out correctly, so you don't need to change anything.

- The **"Default value" text box** lets you preload a form field with information. It's actually the same as a text field's *initial value*, as described on page 410. You can also add a dynamic value generated by your server model's programming language. If you had a field called *date*, then you could automatically add today's date to the field by typing *<?php echo date(*"Y-m-d") ?> (in the PHP/MySQL server model).

  Depending on what type of field you selected from the "Submit as" menu, the Default value text box might be replaced with one of three different types of controls: For a checkbox, radio buttons provide the simple choice of either checked or not checked; if you selected Menu or Radio Group, a Properties button appears, which lets you set the options that appear in the drop-down menu or radio group on the Insert form—the process is the same as adding a dynamic menu or dynamic radio button group, as described on page 858.

  In most cases, you'll change the label of every column. But for now, leave the other options alone.

7. **Click OK to close the window and create the form.**

   Dreamweaver inserts a table, form, and all of the form elements you specified. At this point, the page is complete and ready to accept information. Unfortunately, Dreamweaver doesn't let you return to this window to make any changes. (If you quickly realize that you made a mistake, you can always use Ctrl+Z [⌘-Z] to undo the operation, and then reapply the Insert Record Form wizard.)

To ensure your form works correctly and doesn't produce any errors when it's submitted, add form validation. The powerful new Spry validation tools make this easy (see page 421).

Warning: Once you've added the form, don't rename any form fields. You'll break the script responsible for inserting the record. If that happens, see page 849 for information on updating the Insert Record server behavior.

Using the Insert Record Behavior

Dreamweaver's Record Insertion Form wizard makes quick work of adding the table, form, and programming code required to create a Web page for adding data to a table. At times, though, you might want a more customized approach. Perhaps you've already designed a form that you'd like to use, or created a beautiful table design for your form. Rather than relying on Dreamweaver's rather pedestrian table and form design, you can supercharge your own design with the Insert Record server behavior.

To build a page for adding database records, start by creating a Web page for your server model (ASP, PHP, ASP.NET, or whatever). Add a form to the page (see Chapter 11). Make sure it has one form field for every column to which you wish to add data. Every time a visitor fills out the form, the database acquires a new record.

In some cases, you don't include certain form fields. For example, a database table's primary key (a unique identifier for each record) is usually generated automatically by the database for each new record. In this case, you wouldn't add a field for this column.

In other cases, you might add *hidden* fields (see page 417) that aren't set by someone filling out the form. Suppose someone signs up for your online newsletter, and you want to store the date of registration. Instead of letting the visitor fill out the date, add a hidden form field with the current date.

Once you've created the form, add the Insert Record server behavior like this:

1. **Choose Window → Server Behaviors to open the Server Behaviors panel.**

   The keyboard shortcut is Ctrl+F9 (⌘-F9). You can also use the Insert Record menu (see Figure 24-3).

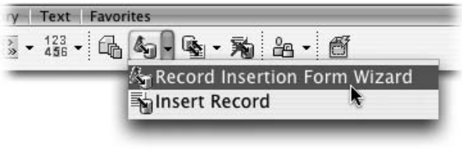


Figure 24-3:
The Insert Record menu (found on the Insert bar's Data tab) lets you use either an automated wizard (Record Insertion Form wizard) or a basic server behavior (Insert Record) to add a record to a database.

2. **Click the + button on the panel, and then select Insert Record.**

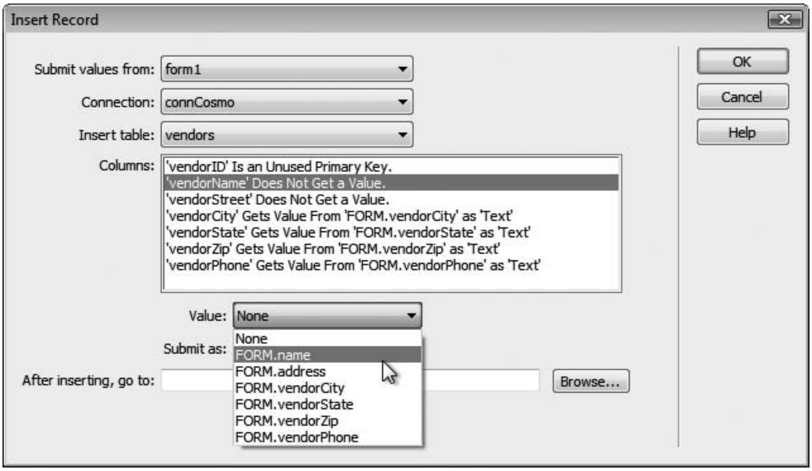
   The Insert Record window opens (see Figure 24-4). It's very similar to the Insert Record Form window, but one key difference is that you must manually associate a form element with the correct database column. (Another difference is that you can't define default values for each form element in this window. You can, however, still apply default values to a form field using the Property inspector, as described on page 857.)

3. **Choose the name of the form that will collect the information for the new record.**

   If there's only a single form on your Web page, you don't have an option here. However, if you have multiple forms (like a search box at the top of the page, as well as the form used to insert a new record), make sure you select the name of the form used for adding data to the database.

4. **Choose a database connection and table.**

   They're the same options described in steps 3 and 4 for the Insert Record Form tool (see page 845). You're telling Dreamweaver which database to use and which table to add data to.



Insert Record

Submit values from: form1

Connection: connCosmo

Insert table: vendors

Columns:
'vendorID' Is an Unused Primary Key.
'vendorName' Does Not Get a Value.
'vendorStreet' Does Not Get a Value.
'vendorCity' Gets Value From 'FORM.vendorCity' as 'Text'
'vendorState' Gets Value From 'FORM.vendorState' as 'Text'
'vendorZip' Gets Value From 'FORM.vendorZip' as 'Text'
'vendorPhone' Gets Value From 'FORM.vendorPhone' as 'Text'

Value: None

Submit as:

After inserting, go to:

Figure 24-4: *The Vendors database table contains a column named vendorName. To ensure that the correct data is stored in the database, from the Value menu, choose the corresponding form field–in this example, FORM.name. (The actual name of the form field is "name;" Dreamweaver, for whatever reason, tacks on FORM at the beginning.) (You may not have to worry about selecting anything; see the Tip below.)*

5. **From the Columns list, select a database column. Choose the name of the form field that collects data for that column.**

   For the form's information to end up in the proper database columns, you must tell Dreamweaver which form field will collect the data for which database column. You can let Dreamweaver choose the proper type from the Submit As pop-up menu. As with step 6 on page 846, this choice depends on how your database is set up (which Dreamweaver can figure out).

Tip: You need to perform step 5 only if the names of the form fields *differ* from the names of the columns in the database. If you name a form field *productName*, and there's also a database column named *productName*, Dreamweaver automatically connects the two in the Insert Record window.

6. **Click the Browse button, and then select a Web page.**

   This step is the same as step 5 on page 846, and indicates which page someone sees after adding a new record to the database.

7. **Click OK to close the window and apply the server behavior.**

   The page is now capable of adding information directly to the database.

If you change the name of a form field, add a form field, or wish to change any of the settings for this behavior, you can edit the Insert Record server behavior by double-clicking Insert Record on the Server Behaviors panel. This selection opens the Insert Record window once again.

Updating Database Records

Maybe someone made an error while entering data into the database. Maybe the price of your product changes. Or maybe you want to provide a way for your Web visitors to update their email addresses. In any case, the time will come when you have to edit an online database.

Creating *Update Record Forms* in Dreamweaver is very similar to creating Insert Record Forms. Both require an HTML form that your audience can fill out and submit. The primary difference is that an update form is *already* filled out with information from the database. It's like a combination of an Insert Record Form and a record detail page (such as the one created by the Master Detail Page set described on page 814).

The first step in creating an update form is to add a *recordset* to the update page (page 782). The recordset will retrieve the data that appears in each field of the update form.

The recordset should contain only a single record—the one to be updated. Therefore, you must use a form or URL parameter to filter the recordset (see page 785 for more on filtering). For example, on a page listing company employees, you could add an Edit button that would link to the page containing the update form. If you added a Go To Detail Page server behavior (see page 818) to the employee listing page, you could then pass the ID for a specific employee to the update page. In turn, the update page would use that ID to filter the database for a single record, and display the employee information in the form. (If all of this sounds confusing on paper, try the tutorial starting on page 867, which takes you step by step through the process.) After you add the recordset to the update form page, you have two options for building an update form. You can either let Dreamweaver automate the process with its Insert Update Record Form wizard, or build a form yourself, and then add the Update Record server behavior. The following pages cover both methods.

The Update Record Form Wizard

Dreamweaver can automate most of the steps involved in creating an update form:

1. **Open a dynamic page with a recordset already added.**

   Remember, the recordset should contain only a single record—the one to update. So you must add a filter when you create the recordset (see page 787 for more on filtering recordsets).

2. **In the document window, click where you wish the insertion form to appear. Choose Insert → Application Objects → Update Record → Record Update Form Wizard.**

   You can also, from the Update Record menu on the Insert bar's Data tab, select Record Update Form Wizard (Figure 24-5). Either way, the Record Update Form window opens (Figure 24-6).

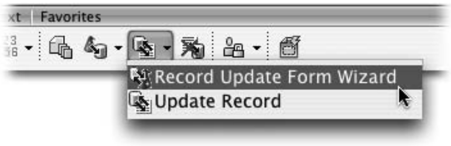
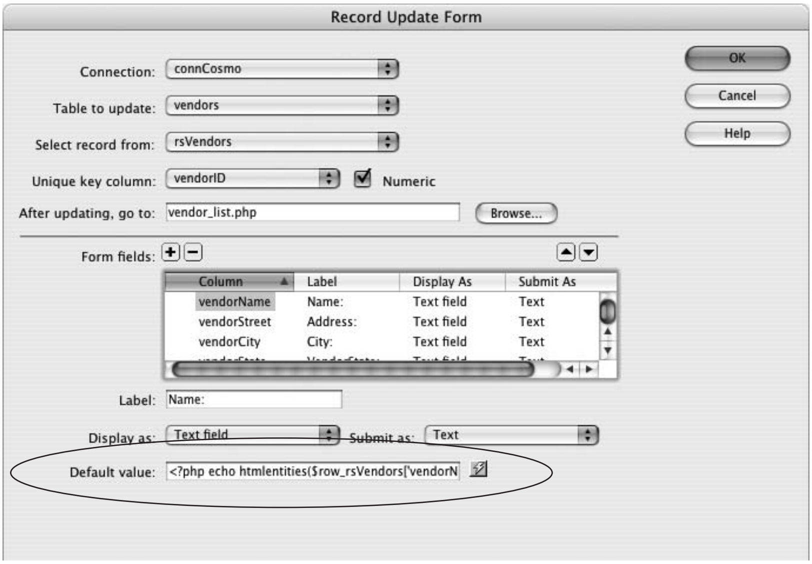


Figure 24-5:
The Update Record menu's wizard can create a no-frills Update Record Form.



Record Update Form

Connection: connCosmo

Table to update: vendors

Select record from: rsVendors

Unique key column: vendorID ☑ Numeric

After updating, go to: vendor_list.php

Form fields:

Column	Label	Display As	Submit As
vendorName	Name:	Text field	Text
vendorStreet	Address:	Text field	Text
vendorCity	City:	Text field	Text

Label: Name:

Display as: Text field Submit as: Text

Default value: <?php echo htmlentities($row_rsVendors['vendorN

***Figure 24-6:** A form field's default value is the information visible when the form first appears on the Web page. In the case of an update form, each form element is already filled out with information from the database. After all, this page is meant for editing data that already exists, not for adding a new record. When using the Update Record Form wizard, Dreamweaver is smart enough to fill out the "Default value" (circled) for each form field using the proper information from the database.*

3. **From the Connection menu, select the database connection.**

   If your site works with several databases, select the connection for the database to which this form will add information.

4. **From the "Table to update" menu, choose a table that matches the recordset you created in step 1.**

   A list of all tables available in the database appears in this menu. You can edit data only from a single table at a time, so select the table that matches the filtered recordset.

Tip: You can include multiple update forms on a single page, so it's possible to create one page that lets you update all the different tables in a database. You'll need to add filtered recordsets for each table you wish to update, and provide links that identify which table you're updating. In one case a URL may include the primary key for one table (update.php?productID=2, for example). Meanwhile, another link passes a different primary key (update.php?categoryID=1) to the same page in order to update a record in a different table.

One way to keep a page like this from appearing overly cluttered with forms is to use the "Show if Recordset Not Empty" server behavior (see page 908). Select one update form, apply the server behavior, and in the "Show if Recordset Not Empty" dialog box, select the recordset that matches the update form. In this way, the page only ever displays one form—the one used to update the record from the specified recordset.

5. **In the "Select record from" menu, make sure the recordset you created earlier is selected.**

   The recordset should be selected already. If it's not, you must have created another recordset first.

6. **In the "Unique key column" pop-up menu, make sure the table's primary key column is selected. If the column contains a number—and it usually does—verify that the Numeric checkbox is turned on.**

   The "Unique key column" is used to identify which record to change during the update process. (For a description of primary keys, see page 769.)

7. **Click the Browse button; navigate to the Web site, and then select a file.**

   This file represents the page that will appear after a record has been updated. A good technique is to select a page that lists the details of the record that's being edited (like the detail page in the master/detail pages discussed on page 814).

   Then, after updating the page, the visitor immediately sees a page showing the changes.

8. **If you like, change a column's settings.**

   After clicking a row in the Form fields list, you can change the Label, Display As menu, and Submit As properties of the database column, the same as when inserting a record as described on page 846. If the table includes a Description column that might hold a fair amount of text, then select Text Area from the Display As menu. This way, the form includes a larger text box that can display all of the description.

   Don't change the Submit As option, which determines what kind of data to submit—text, a number, and so on—because Dreamweaver automatically gets this information from the database. In addition, you would rarely change the Default Value property. Since this represents the data pulled out of the database for a particular record, modifying the default value *always* changes that information in the database.

Note: Sometimes you might want to change the Default Value for a column. Say a table includes a column to track the date a record was last updated. In this case, you would make the Default Value show the current date (instead of the date already stored in the database). When someone updates a record, the database automatically records the current date in the "last update" field.

   To delete a field so it doesn't appear in the form, click the Remove (minus sign [–]) button. For example, you might want to remove the primary key column—the column you selected in step 5. Since the key identifies the record, letting someone edit this could cause problems to your database, such as overwriting data for another record. Fortunately, Dreamweaver is aware of this potential problem; when you use the Update Record Form wizard, Dreamweaver doesn't create an editable form field for the primary key. Instead it simply displays the number on the page for your reference.

9. **Click OK to close the window.**

   Dreamweaver inserts a table, form, and all the form elements you specified.

Note: If you get the following error–"Please choose a unique key from the selected Recordset, or Click Cancel"–then the recordset you added to the page didn't include the table's primary key. You should cancel the Update wizard, edit the recordset, and then click the "All" button next to the word Columns. Then reapply the Record Update Form wizard.

Once you click OK to close the Record Update Form dialog box, you can never again return to it. From now on, you make any changes by editing the Update Record server behavior, as described next. You're also free to use any of Dreamweaver's editing tools to format the table, labels, and form elements any way you wish.

Note: After using the Update wizard, don't change the name of the form or its fields. Because the program that updates the database relies on these names, changing them stops the update code from working.

The Update Record Server Behavior

Dreamweaver's Record Update Form wizard makes it delightfully easy to add the table, form, and programming code required to create a Web page for editing database records. But when you need more flexibility—if you've already designed a form that you'd like to use or created a table design for your form—the Update Record server behavior lets you keep your own beautiful design and give it the power to update a database.

You must start with a page that has a filtered recordset, as described on page 849. Then add a form to the page (see Chapter 11). Make sure the form has one form field for every database column to be edited. Don't include a form field for the database table's primary key. (Allowing anyone to change the primary key could have disastrous effects on your database.)

Tip: Giving your form fields the same names used for the database table's columns speeds up the process of adding the Update Record server behavior.

At this point, the form is full of empty fields. If you preview the page, none of the data you retrieved from the recordset appears. To fill the form with data, you must *bind* data from the recordset to each form field, as follows:

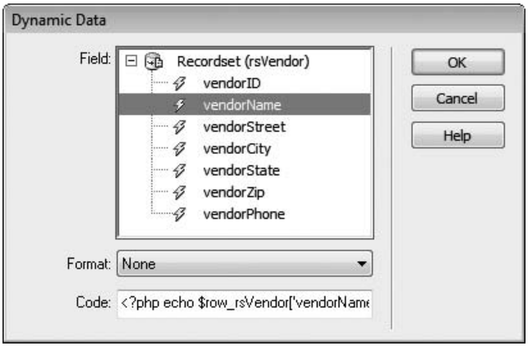
1. **In the document window, select the form field.**

   Just click the form field to select it.

Note: These instructions apply to text and hidden fields. Information on binding data to radio buttons, checkboxes, and menus appears on page 858.

2. **In the Property inspector, click the dynamic data button (the lightning bolt) to the right of the Init Val box.**

   The Dynamic Data window appears (Figure 24-7).



Dynamic Data

Field:
- Recordset (rsVendor)
  - vendorID
  - vendorName
  - vendorStreet
  - vendorCity
  - vendorState
  - vendorZip
  - vendorPhone

Format: None

Code: <?php echo $row_rsVendor['vendorName

Figure 24-7:
The Dynamic Data window lets you add information from a recordset to form fields, so parts of a form can be pre-filled out. This ability is exactly what you want when updating information already in the database. In fact, the Dynamic Data window displays and lets you use additional data sources (like URL variables) that you've added to the page (see page 897).

Tip: You can also bind data from a recordset to a form field by dragging the database column from the Bindings panel (see page 801), and then dropping it onto the form field.

3. **Click the + button next to the recordset used for the update page.**

   A list of all columns retrieved by the recordset appears.

4. **Select the name of the table column you wish to bind to the form field. Click OK.**

   You should ignore the Format menu and Code box. The form field is now set to display information from the recordset.

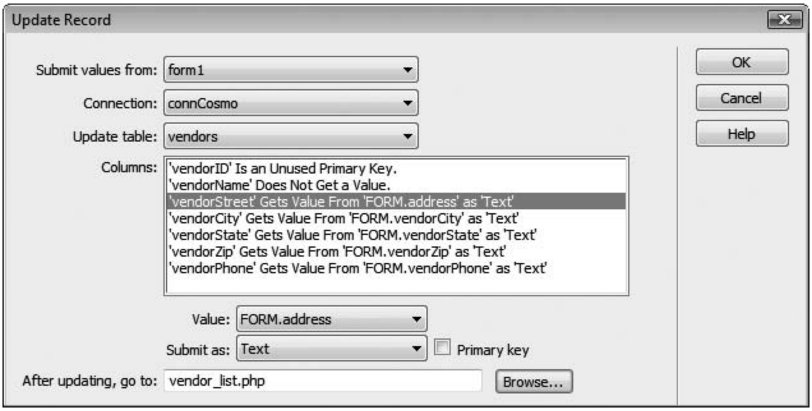
Once you've created the form and added recordset information to each field, add the Update Record server behavior:

1. **Choose Window → Server Behaviors to open the Server Behaviors panel.**

   The keyboard shortcut is Ctrl+F9 (⌘-F9).

2. **Click the + button on the panel, and then select Update Record.**

   The Update Record window opens (see Figure 24-8).



Update Record

Submit values from: form1

Connection: connCosmo

Update table: vendors

Columns:
```
'vendorID' Is an Unused Primary Key.
'vendorName' Does Not Get a Value.
'vendorStreet' Gets Value From 'FORM.address' as 'Text'
'vendorCity' Gets Value From 'FORM.vendorCity' as 'Text'
'vendorState' Gets Value From 'FORM.vendorState' as 'Text'
'vendorZip' Gets Value From 'FORM.vendorZip' as 'Text'
'vendorPhone' Gets Value From 'FORM.vendorPhone' as 'Text'
```

Value: FORM.address

Submit as: Text □ Primary key

After updating, go to: vendor_list.php

Figure 24-8:
The Update Record window is very similar to the Record Update Form window (Figure 24-6). The main difference is that here you must manually associate a form element with the correct database column.

3. **From the "Submit values from" menu, select the name of the form.**

   It's possible to have multiple forms on a single page: a form for editing a record and a form for searching the site, for example. Select the name of the form used for editing data.

4. **Select the database connection and the table you want to update.**

   They're the same as described in steps 3 and 4 for the Record Update Form wizard on page 850. They tell Dreamweaver which database to use, and which table to update data to.

5. **Select a form element from the list, and then, from the Column menu, select the matching database.**

   For the form's information to end up in the proper database columns, you must tell Dreamweaver which database column you want to store the selected form field's information.

Tip: If you use the name of your database columns to name the form fields too, Dreamweaver automatically associates the two, so you can skip this step.

   From the Submit As menu, let Dreamweaver choose the proper type. As with step 6 on page 846, this choice depends on how your database is set up (which Dreamweaver can figure out).

6. **Click the Browse button; navigate to the Web site, and then select a file.**

   This file represents the page that appears after a record has been updated. A good technique is to select a page that lists the details of the record that's being edited (like the detail page in the master/detail pages, discussed on page 814).

7. **Click OK to close the window and apply the server behavior.**

   The page is now capable of editing information from the database.

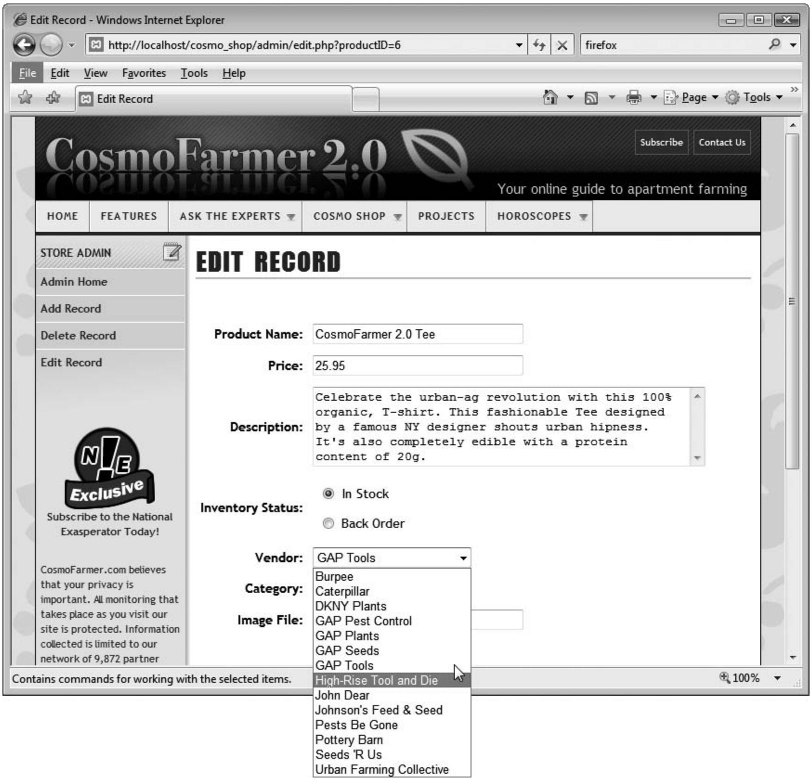
As with any server behavior, double-clicking its name on the Server Behaviors panel—in this case, Update Record—opens the behavior's dialog box so you can make changes to its properties.

Dynamic Form Fields

Form fields don't necessarily have to be empty. For example, the record update forms covered in the previous section are already filled out with data pulled from the database. Likewise, you don't necessarily have to manually create the options in a pull-down menu; they could come from information in a database.

Imagine a scenario where you create an Employee Directory section for your company's Web site. On the Insert New Employee page, you build a pop-up menu that lets the Human Resources department select a department for the new employee.

You, the designer, *could* create a menu like this department list by manually typing the name of each department. But what if the names of the departments change, or new departments are added? You'd have to reopen the page and edit the form field each time. But if you opted for a dynamic menu instead, the page would build the Departments pop-up menu automatically by retrieving the current list of departments from the database. Figure 24-9 shows a similar example of a form pull-down menu that's been dynamically created.


Edit Record - Windows Internet Explorer
http://localhost/cosmo_shop/admin/edit.php?productID=6
firefox
File Edit View Favorites Tools Help
Edit Record
Page Tools
CosmoFarmer 2.0
Subscribe
Contact Us
Your online guide to apartment farming
HOME
FEATURES
ASK THE EXPERTS
COSMO SHOP
PROJECTS
HOROSCOPES
STORE ADMIN
Admin Home
Add Record
Delete Record
Edit Record
N!E
Exclusive
Subscribe to the National Exasperator Today!
CosmoFarmer.com believes that your privacy is important. All monitoring that takes place as you visit our site is protected. Information collected is limited to our network of 9,872 partner
Contains commands for working with the selected items.
100%
Burpee
Caterpillar
DKNY Plants
GAP Pest Control
GAP Plants
GAP Seeds
GAP Tools
High-Rise Tool and Die
John Dear
Johnson's Feed & Seed
Pests Be Gone
Pottery Barn
Seeds 'R Us
Urban Farming Collective

EDIT RECORD

Product Name: CosmoFarmer 2.0 Tee

Price: 25.95

Description: Celebrate the urban-ag revolution with this 100% organic, T-shirt. This fashionable Tee designed by a famous NY designer shouts urban hipness. It's also completely edible with a protein content of 20g.

Inventory Status: ● In Stock ○ Back Order

Vendor: GAP Tools

Category:

Image File:

Figure 24-9:
Dynamic form fields come in handy with update forms. Form fields are already filled out with database information that's ready to be edited. Menus can also be dynamically generated from records in a recordset. In this case, the menu (shown open) lists records retrieved directly from a database table containing the product categories for the CosmoFarmer online store.

In essence, a dynamic form field is a form element whose value, labels, or other settings come from dynamic data in the Bindings panel. The dynamic data can come from a recordset (as with an update form), a form or URL parameter, or even a cookie or session variable (see page 901).

Whenever you wish to use a dynamic form field, start by creating a form. Add all form fields that might include dynamic content. (Not all the fields have to be dynamic, however. In the employee directory example discussed earlier, only the Department menu on the Insert New Employee form would be dynamic. The other fields for entering an employee's information would be empty.)

Next, add a recordset, request variable, session variable, or application variable to the Bindings panel (see page 897 for information on those last three choices). Then, finally, attach the dynamic data to the form field. The process for binding dynamic data depends on the type of form field.

Dynamic Text Form Fields

Any form field that accepts typing—text, text area, and password fields—can be dynamic. For example, if a site requires a user login, you could include a "remember me" feature, so that when a visitor who's previously signed in returns to the site, the user name and password are already filled out.

Tip: If you like the idea of a "remember me" feature for password-protected pages, try the free Save Password Login Form extension, which works with ASP, ASP.NET, PHP, and ColdFusion. You can find it on the Adobe Exchange Web site (see page 738).

You can add dynamic data to a text field (also called *binding* data to the field) using any of the methods described below. (Remember, you must first have added the form field to the page and added the dynamic data to the Bindings panel, which means adding a recordset to a page, as described on page 782, or creating additional data sources [like cookies or session variables], as described on page 897.)

- In Design view, drag the dynamic data item from the Bindings panel, and drop it onto the form field.
- In Design view, select the text field. In the Bindings panel, select the dynamic data item, and then click the Bind button.
- Select the text field. In the Property inspector, click the dynamic data button (the lightning bolt). The Dynamic Data window appears (see Figure 24-7); select the dynamic data item from the list, and then click OK.
- In the Server Behaviors panel, click the + button, and then select Dynamic Form Elements → Dynamic Text Field. In the window that appears, you see a text field menu. Select the text field to which you wish to add dynamic data; then click the lightning bolt button to open the dynamic data window. Select the dynamic data item from the list, and then click OK. (Insert → Data Objects → Dynamic Data → Dynamic Text Field works, too.)

Note: You can bind dynamic data to a *hidden* field (page 417) using the same steps.

After binding the data to the field, the name of the data item appears inside the field—{rsDetails.adName}, for example. If you're using Live Data view (page 812), the actual data from the database appears inside the field.

Note: Dreamweaver lets you format dynamic data in a form field just like dynamic text you add to a page, as described on page 802. Be careful with this option, though. If the form is submitting to a database, data may sometimes have to be in a certain format. For example, prices are often stored as numbers inside a database. But using Dreamweaver's currency format, you can make a price appear as $34.00 in a form field. When your visitor submits the form, the $ sign goes along for the ride, causing the database to spit out a horrible error message upon encountering this nonnumeric character.

To remove dynamic data from a text field, just select the field, and then, in the Bindings panel, click the Unbind button. (Deleting the contents of the field's Init Value box in the Property inspector also works.)

Dynamic Checkboxes and Radio Buttons

With a text field, you can dynamically change the *value* of the field. With checkboxes and radio buttons, however, you can control only their status (checked or unchecked) dynamically.

You can use this value to select one radio button in a group based on a value in the database. As part of a product ordering system, shoppers could select a particular shipping option: USPS, FedEx, or UPS. But after reviewing her orders, what if someone changes her mind and chooses a different shipping option? When she returns to the order page, you'd want the Shipping Option radio button to reflect the choice she had made earlier. In other words, you want the page to read the shipping option for the order from the database, and highlight the radio button that matches. (See an example of this in the "Building a Page for Editing Database Records" section of the tutorial starting on page 875.)

Dynamic radio buttons

You add dynamic radio buttons like this:

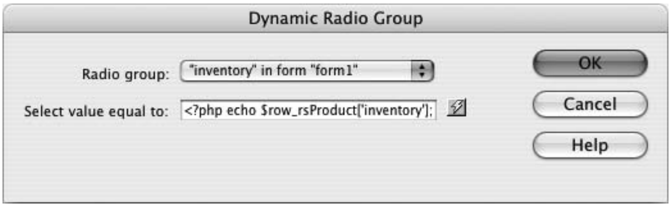
1. **Add a group of radio buttons to the page.**

   You should have as many radio buttons as there are possible values stored in the database column. Remember, if you wish to create a group of related radio buttons, you must give every button in the group the same name (see page 411).

   Note, too, that the value of each radio button must also exactly match the values stored in the database. If a Shipping column in the database stores *USPS*, *FedEx*, or *UPS*, then the radio group should have three buttons. Each button would share the same name—*shipping*, for instance—but their checked values would match the different values stored in the database: USPS, FedEx, and UPS. Capitalization counts, so if the value in the database is UPS, the radio button value must be UPS, not Ups, ups, or UpS.

2. **Open the Server Behaviors panel (Window → Server Behaviors). Click the + button, and then select Dynamic Form Elements → Dynamic Radio Group.**

   The Dynamic Radio Group window appears (see Figure 24-10).



Radio group: "inventory" in form "form1"

Select value equal to: <?php echo $row_rsProduct['inventory'];

Figure 24-10: *The Dynamic Radio Group window lists the values of each button in the group. By selecting a button from the list, you can change its value in the Value field.*

3. **From the first menu, choose the radio button group.**

   If your form has more than one group of radio buttons, select the one you wish to be dynamic.

4. **Click the dynamic data button (the lightning bolt) to the right of the "Select value equal to" field.**

   The Dynamic Data window opens (see Figure 24-7). Select the dynamic data item for this radio group. In a nutshell, the radio button whose value matches the dynamic data is selected. If no radio buttons contain the same value, then no buttons are selected.

Note: You set the "Select value equal to" field only once per radio group (not once per button in the group).

5. **Click OK to close the window.**

   Dreamweaver adds a Dynamic Radio Buttons server behavior to the page.

Dynamic checkboxes

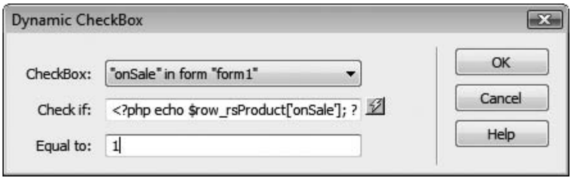
Dynamic checkboxes work almost the same way:

1. **Add a checkbox to the page.**

   This process is described on page 411.

2. **Open the Server Behaviors panel (Window → Server Behaviors); click the + button, and then select Dynamic Form Elements → Dynamic CheckBox.**

   The Dynamic CheckBox window appears (Figure 24-11).



CheckBox: "onSale" in form "form1"

Check if: <?php echo $row_rsProduct['onSale']; ?

Equal to: 1

Figure 24-11: *The Dynamic CheckBox server behavior lets you control whether or not a checkbox is turned on, based on information from a database, cookie, URL parameter, or other piece of dynamic data.*

3. **If the form has more than one checkbox, select a checkbox from the first menu.**

   Select the checkbox you wish to control dynamically.

4. **Click the dynamic data button (the lightning bolt) to the right of the "Check if" field.**

   The Dynamic Data window opens (see Figure 24-7). Select the dynamic data item for this checkbox.

5. **Type a value into the "Equal to" box.**

   If the value from the dynamic data (previous step) matches the value you provide here, the checkbox is turned on. (If the checkbox is part of an update form, this should match the value you gave the checkbox from step 1.)

6. **Click OK to close the window.**

   Dreamweaver adds a Dynamic CheckBox server behavior to the page.

To *remove* the dynamic properties from a group of radio buttons or a checkbox, open the Server Behaviors panel (Window → Server Behaviors). Among the list of server behaviors, you see the dynamic radio button or checkbox behavior. It looks something like "Dynamic Radio Buttons(group_name)" or "Dynamic Check Box(checkbox_name)"—where *group_name* or *checkbox_name* is replaced with whatever name you gave the buttons or checkbox. Select it, and then click the Remove (minus sign [–]) button. (Pressing the Delete key does the same thing.)

Dynamic Menus and Lists

Dynamic menus and lists are among the commonest form elements. You can use them for more than just update forms. Even a form used to add information to a database can use a dynamic form to display a list of options stored in the database. They save you the effort of having to rebuild traditional menus or lists every time your company opens a store in a new state, adds a new employee department, or adds a new category to its product line.

To create a dynamic menu or list, proceed as follows:

1. **Create a form, and then add a menu or list to the page.**

   For example, choose Insert → Form, and then Insert → Form Objects → List/Menu. This process is the same as adding a menu or list, as described on page 414. Don't add any items to the list, though. That's the whole point of this exercise: Dreamweaver builds the menu or list automatically.

2. **Add a recordset to the page, which includes the information you wish to appear in the menu or list.**

   Perhaps you want to create a menu listing the different categories of products your company sells—books, DVDs, CDs, lederhosen, clogs, and so on. If the database has a table containing the categories, you could then create a recordset that retrieves the name of each category. (In most cases, you also retrieve a primary key, like the category ID field. The name of the category appears in the menu, while the primary key is the value submitted with the form.)

3. **In the document window, select the menu or list, and then, on the Property inspector, click the Dynamic button.**

   The Dynamic List/Menu window opens (see Figure 24-12). The name of the menu or list you selected appears in the Menu box.

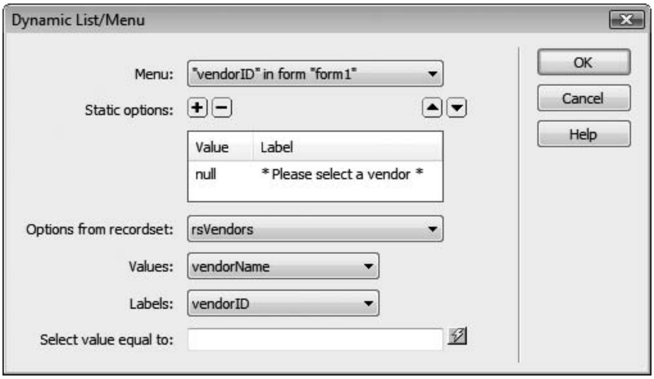


Figure 24-12:
Dreamweaver simplifies the process of creating automatically generated menus like the one pictured in Figure 24-9. Using information pulled from a database, you never have to create another menu by hand.

4. **If you like, add some static options to the menu or list.**

   A *static option* is simply a value and a label that you enter by hand—menu or list items, appearing at the top of the list, that don't change. This option doesn't come from the recordset you created.

   You could use this feature for options not likely to change—Amazon.com may begin to sell electronics, hors d'oeuvres, and dermatology services, but chances are good that books will always be among its categories. You can also use it to provide clear instructions about operating this menu: "Pick a state," or "Choose an option from this list," for example. Of course, this step is optional. (Adding static options works just like it does in a regular menu's List Values box, as described on page 414.)

5. **From the Options From Recordset menu, choose the recordset you created in step 2.**

   You've just told Dreamweaver where the items to be listed in the menu are coming from.

6. **From the Values menu, choose a table column; choose another column from the Labels menu.**

   Menu and list items consist of a *label* (what someone actually sees in the menu) and a *value* (the information that's transmitted when the form is submitted). Using the example in step 1, you would select *categoryID* (or whatever is the name of the recordset's primary key) from the Value menu, and *categoryName* from the Label menu. If the label and value are the same, choose the same table column from both menus.

7. **If you want your menu to have one item preselected, then click the dynamic data button (the lightning bolt). In the Dynamic Data window (Figure 24-7), select a dynamic data item.**

   This step is optional and most frequently used for an update form. Suppose you create a form for updating information on your catalog of products. When the update page loads, all the form fields would already be filled out with information about a particular product. The menu that lets you specify the product's category, therefore, should have the name of the category that matches the database record displayed and preselected.

8. **Click OK.**

   Dreamweaver adds a Dynamic List/Menu server behavior to the page.

You can remove the dynamic menu or list by selecting and deleting it. To leave the menu but remove its *dynamic* properties, open the Server Behaviors panel (Window → Server Behaviors). Here, you see the dynamic list/menu behavior; it's listed as "Dynamic List/Menu(menu_name)," where *menu_name* is the name you gave the menu or list. Select it, and then click the Remove (minus sign [–]) button or press the Delete key.

To edit the behavior, just select the menu in the document window, and then, once again, in the Property inspector, click the Dynamic button.

Deleting Records

Dreamweaver's Delete Record server behavior lets you build pages that let people remove records from a database. Depending on which server model you use, the method of adding this server behavior varies. The Dreamweaver server behavior for ASP and JSP is the same, while the server behavior for PHP, ASP.NET, and ColdFusion offers a bit more flexibility.

Deleting Records for ASP and JSP

The setup is similar to using the Update Record server behavior, in that you must create a page with a recordset that retrieves a single record—the item to delete. This page is just like a detail page, as described on page 414. Another page must link to the delete page, and provide the proper key for filtering to a single record on the delete page.

One way you can do this is by adding a link—Delete This Record, perhaps—on a record detail page. The link should pass the primary key for the condemned record to another page (the "Go to Detail Page" server behavior described on page 818 can help with this). This page—the delete page—would include a recordset that retrieves the file to be deleted. To confirm the deletion, the page might say something like, "Are you sure you wish to delete this record?" (Adding some dynamic text from the recordset, such as a name column, will help identify the record.)

A delete page differs from a regular detail page because it includes a form with a single Submit button, and a Delete Record server behavior.

To create a delete page, proceed as follows:

1. **Create a dynamic page containing a filtered recordset.**

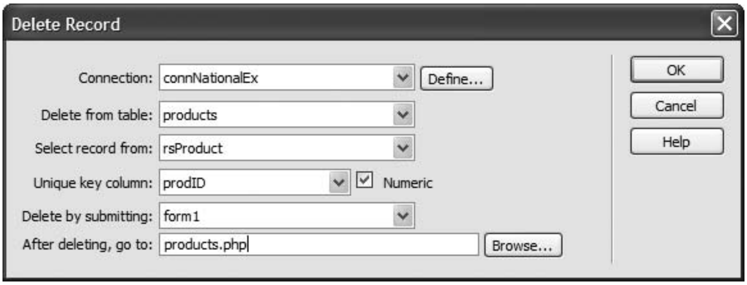
   As noted above, the recordset should retrieve a single record—the one to be deleted. However, you don't need to retrieve *all* the columns for the record. At a minimum, the recordset must retrieve the record's primary key, since the Delete Record server behavior needs it to know which record to remove. But beyond that, you should probably include some identifying information on the delete page, so that your visitor can make sure she's really deleting the right record. If the page deletes an employee from the database, consider putting the employee's name on the page as one final check.

2. **Add a form to the page with a single Submit button (see page 401).**

   Change the label of the button so that it reflects the action it will perform—Delete This Record, for example.

3. **Open the Server Behaviors panel (Window → Server Behaviors). Click the + button, and then select Delete Record.**

   The Delete Record window appears (see Figure 24-13).



Connection: connNationalEx

Delete from table: products

Select record from: rsProduct

Unique key column: prodID ☑ Numeric

Delete by submitting: form1

After deleting, go to: products.php

Figure 24-13: *Dreamweaver's Delete Record server behavior simplifies the process of removing information from a database.*

4. **From the Connection menu, select a database. From the "Delete from table" menu, select the table with the record to be deleted.**

   These steps should be familiar by now; the table you select should be the same one you used when creating the detail recordset for this page.

5. **From the "Select record from" menu, select the recordset you added to this page.**

   This step specifies the recordset on the page (if there's more than one) that Dreamweaver should use for specifying the record to be deleted.

6. **From the "Unique key column" menu, choose the table's primary key. If it's a number, make sure the Numeric box is also turned on.**

The "Unique key column" identifies which record to delete. (For a description of primary keys, see page 769.) Make sure you retrieved this primary key information when you created the recordset, or the server behavior doesn't work.

7. **Select the name of the form that contains the Delete button.**

   Unless the page has more than one form, the name of the form you created in step 2 should automatically appear here.

8. **Click the Browse button; navigate to the Web site, and then select a file.**

   This file represents the page that will appear after a record has been deleted. This could either be a page with a confirmation message ("The Record has been successfully deleted") or a page that lists the remaining records in the database.

9. **Click OK.**

   Dreamweaver inserts the code necessary to delete a record from the database.

Deleting Records for PHP, ASP.NET, and ColdFusion

Deleting a record using ASP or JSP is straightforward. For the other server models, the Delete Record server behavior offers a bit more flexibility, and you can implement it in a variety of ways. You don't have to add a recordset to a page to delete a record, nor do you have to add a form with a delete button.

The main requirement: The page with the Delete Record server behavior must have some way to retrieve the primary key for the record you wish to delete. This can be a form, a URL, a cookie, a session variable, or any of the other data sources discussed on page 897.

On a page that lists the details of a particular record, you could include a link to a delete page and pass the ID of that record in the URL (see "Passing Information Between Pages" on page 817). The delete page could then use the ID number in the URL to delete the record, and then send the visitor off to another page—perhaps a page verifying that the record was successfully deleted.

Note: Because of this flexibility, under the PHP, ASP.NET, and ColdFusion server models, you could place a Delete Record server behavior on a blank dynamic page. All the page would do is delete the specified record, and then go to another page on the site.

There are many ways, therefore, that you could delete a record in this server model. Here's a method that would provide the same experience as the ASP and JSP models discussed above—that is, a delete page that lets people confirm the item they wish to delete by clicking a button on a form:

1. **On one of the pages in your site, add a link to the delete page.**

   On a page that provides the details of a single record, you could add a link to the delete page—maybe the word "Delete" or a button with a picture of a trash can.

Alternatively, you could add a “delete this record” link as part of a repeating region (see page 805). In this way, you would have multiple records on a single page, each with its own link to the delete page. In both cases, you’d attach the record’s primary key to the link, as described on page 817.

Note: Unfortunately, Dreamweaver doesn’t provide a tool for deleting more than one record at a time.

2. **Create a dynamic page—the delete page—containing a filtered recordset.**

   This recordset should retrieve a single record—the one to be deleted. As mentioned for the ASP and JSP models, you don’t need to retrieve *all* the columns for the record. At a minimum, the recordset must retrieve the record’s primary key, since the Delete Record server behavior needs it to know which record to remove. You may want to include some identifying information, such as the name of the item to be deleted, so that your visitors can see what they’re about to delete.

3. **Add to the page a form consisting of a single Submit button (as described on page 418).**

   When you create the button, change its label to reflect what it does—Delete This Record, for example.

4. **Select the form. Then, in the Property inspector’s Action box, type the page’s file name.**

   If the page is called *delete.php*, type *delete.php*.

   The *Action* property indicates where the form should be sent (see page 402). In this case, when the visitor clicks the Submit button, the form information goes *back* to the same page.

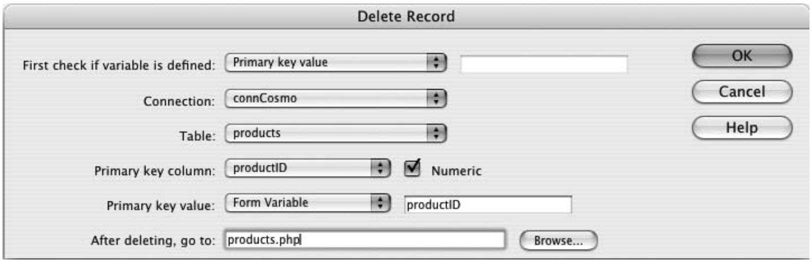
   This kind of trickery is common in dynamic pages. When your visitor clicks the Delete button, the form is sent to the same page—but this time the form doesn’t show up. Instead, the Delete Record server behavior (which you’re going to add in step 6) deletes the record, and then sends the visitor off to another page.

5. **Add a hidden field to the form (page 417). Name this field whatever you wish, but bind (attach) to it the primary key from the recordset you created in step 2.**

   This hidden field is what tells the Delete Record server behavior which record to delete. (For instructions on binding dynamic data to a form field, see page 855.)

6. **Open the Server Behaviors panel (Window → Server Behaviors). Click the + button, and then, from the list of server behaviors, select Delete Record.**

   The Delete Record window appears (Figure 24-14). This window differs slightly among ASP.NET, PHP, and ColdFusion. However, the basic steps described here are the same.



Delete Record

First check if variable is defined: Primary key value

Connection: connCosmo

Table: products

Primary key column: productID ☑ Numeric

Primary key value: Form Variable productID

After deleting, go to: products.php

Figure 24-14: *The Delete Record window for PHP differs slightly from its .NET and ColdFusion counterparts. .NET lets you specify an error page so that visitors see a different page if the delete action fails; ColdFusion lets you specify a user name and password for the database.*

7. **From the "First check if variable is defined" menu, choose "Primary key value."**

   This step may seem like putting the cart before the horse, because you won't define the primary key value until step 10 below. However, this option lets you control *when* the record is deleted, by making sure the proper variable is defined *before* the server behavior deletes the record. In this case, the record doesn't go away until the visitor clicks the Delete button, causing the server to send a form variable containing the record's primary key.

   This precaution is necessary; without this option, the record would be deleted whenever the page loads. Since the page really serves two functions—letting visitors confirm that they wish to delete the record, and actually deleting the record from the database—you need to make sure the visitor has first visited the page, and *then* clicked the Submit button you added in step 3.

8. **From the Connection menu, select a database. From the Table menu, select the table with the record to be deleted.**

   These steps should be familiar by now. The table you select should be the same one you used when creating the detail recordset for this page.

9. **From the "Primary key column" menu, select the table's primary key.**

   This tells Dreamweaver which database field contains the unique key that identifies which record to zap from the table.

10. **From the "Primary key value" menu, select Form Variable. In the box just to the right of that menu, type the name of the hidden field you added in step 5.**

    This step is the final piece of the puzzle. It tells the server behavior where to find the ID number for the doomed record. In this case, the form with its hidden field supplies the ID number.

11. **Click the Browse button; navigate to the Web site, and then select your confirmation page.**

    The confirmation page appears after a record has been deleted. It's a page you've created in advance—either a page with a confirmation message ("The Record has been successfully deleted"), or a page that lists the records of the database—a *master* page.

12. **Click OK.**

    Dreamweaver inserts the code necessary to remove a record from the database.

As with any of the other server behaviors that change content in a database, you should carefully control who has access to the delete page. Going to work one morning and finding that someone has deleted all the products from your company's e-commerce site is enough to ruin your whole day. Dreamweaver's User Authentication behaviors, discussed in the next chapter, can help.

If you ever need to change any of the properties of the Record Delete action, such as picking a different page to go to after a record is deleted, you can go to the Server Behaviors panel, and then double-click Delete Record. The Delete Record window opens; make any changes, and then click OK.

Tutorial: Inserting and Updating Data

In this tutorial, you'll continue working on CosmoFarmer's online store. You'll work on two administrative pages that allow employees of CosmoFarmer to add new products to the database, and to edit products already in the database.

This tutorial assumes you've already completed the tutorials for Chapters 22 and 23. If not, turn to page 754, follow the instructions for preparing the application server, database, and Dreamweaver for this project. Then turn to page 821 and build the product catalog pages.

Adding an Insert Product Page

Start by opening a page that's already been created:

1. **Open the file named *add.php* in the admin folder of the local site you defined in Chapter 22.**

   This page contains three Spry tabbed panels like the ones you learned about on page 458. Instead of having several Web pages, each dedicated to inserting one record into one database table, you can collect all the insert forms on a single page. Clicking a tab opens the tabbed panel with the appropriate insert form.

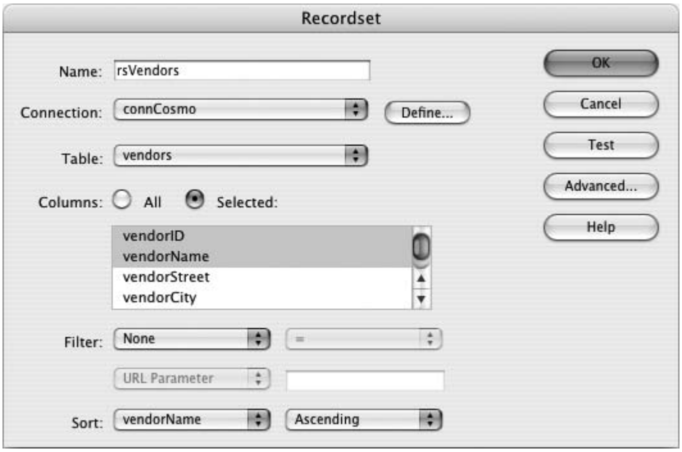
   You'll also notice that this page is stored inside a folder named *admin.* Pages for adding and editing the online store's products shouldn't be accessible to the public; you wouldn't want just anyone adding products—"The Electric Whoopee Cushion, by Mr. Hacker," for example—to the store. Accordingly, these pages are kept in a folder reserved for administrators of the Web site. (In the next chapter, you'll learn how to password-protect these pages.)

   Each new product requires an ID number for the vendor who manufactures the product. The database for these products actually contains several tables: Products, Vendors, and Category. Information about each vendor (name and contact info) is in the Vendors table, while information on each product (price, description, and so on) is in the Products table. A third table contains a list of product categories, which you used in the last tutorial to create the category navigation bar.

To keep the Vendors and Products tables connected, so that you know which vendor manufactures which product, the Products table includes a field containing the vendor's ID number. Whenever you add an item to the Products table, then, you also need to insert the vendor's ID number. One way would be to have a CosmoFarmer employee type the vendor *number* each time a product is added to the page. But that approach is prone to error, since the employee needs to remember which number belongs to which vendor: "Is Gap Plants vendor 2 or 3?" A better method is to provide a pull-down menu that lists the name of each vendor, but which submits the vendor's number when the product is added to the database. To make this kind of dynamic menu, start by creating a recordset.

2. **Make sure the Bindings Panel is open (Window → Bindings). In the Bindings Panel, click the + button, and then select Recordset.**

   Or use any of the methods described on page 782 to add a new recordset; for example, choosing Insert → Data Objects → Recordsets*;* clicking the + button in the Server Behaviors panel, and then choosing Recordset (Query); or using the Recordset button on the Insert bar's Data tab. Either way, the Recordset window opens. Make sure that the simple recordset options show up (see Figure 24-15). Next, you'll define this recordset's properties.



Recordset

Name: rsVendors

Connection: connCosmo

Table: vendors

Columns: ○ All ● Selected:

vendorID
vendorName
vendorStreet
vendorCity

Filter: None =

URL Parameter

Sort: vendorName Ascending

Figure 24-15:
When creating this recordset, make sure you're using the window's Simple options. If you see a button labeled Advanced, you're in the right place. (If that button's missing, click the Simple button to access the basic recordset options.)

3. **In the Name box, type *rsVendors*. From the Connection menu, select connCosmo. From the Table menu, select "vendors."**

   These three steps set up the name, database, and table required for the recordset. For a recap of creating recordsets, turn to page 782.

4. **Click the Selected radio button; from the Columns list, select vendorID and vendorName.**

   You can do this step by holding down the Ctrl (Option) key while clicking the name of each column. Finally, pick an order for sorting the list of vendors.

5. **From the Sort menu, choose vendorName. Make sure that in the Order menu, Ascending is selected.**

   The Recordset window should now look like Figure 24-15.

6. **Click OK to close the window and insert the recordset in the page.**

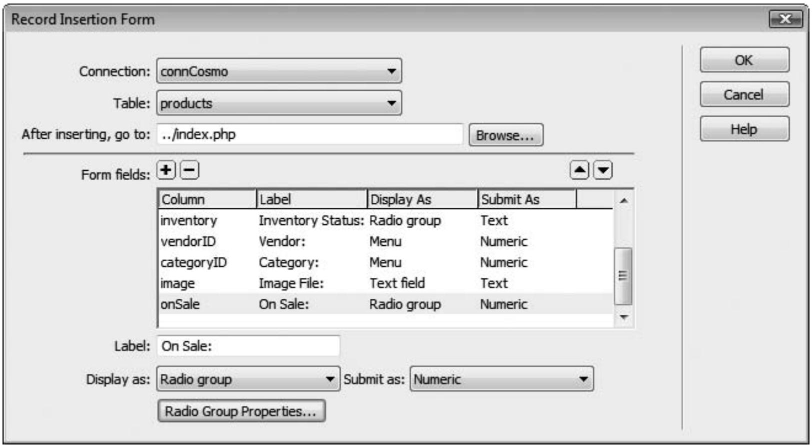
   Each product also belongs to a particular category. In the Products table in the database, the category is represented by a number, but the category names are stored in the Categories table. This situation is the same as vendors, so you'll create a dynamic menu for inserting the proper category name.

7. **Add another recordset to the page by following steps 2–6: Name the recordset rsCategories, select the "category" table, choose the All columns radio button, and then sort by CategoryName.**

   You've just added a second recordset to this page. Now you're ready to add a form for inserting a new record. The page contains three Spry tabbed panels. You'll add a form for inserting new products into the database via the tab that's visible when you open the page—the Add Product tab.

8. **Click the empty area of the gray box directly below the Add Product tab. Choose Insert → Data Objects → Insert Record → Record Insertion Form Wizard.**

   (Other methods of inserting this wizard are discussed on page 844.) The Record Insertion Form window opens (Figure 24-16). Next, you'll tell Dreamweaver which database to connect to, and which table will receive data from the form.



Connection: connCosmo

Table: products

After inserting, go to: ../index.php

Column	Label	Display As	Submit As
inventory	Inventory Status:	Radio group	Text
vendorID	Vendor:	Menu	Numeric
categoryID	Category:	Menu	Numeric
image	Image File:	Text field	Text
onSale	On Sale:	Radio group	Numeric

Label: On Sale:

Display as: Radio group Submit as: Numeric

Figure 24-16: *While you can manually create a form and program it to insert a new record in a database, Dreamweaver's Record Insertion Form wizard makes the task a snap.*

9. **From the Connection menu, choose "connCosmo." From the Table menu, choose "Products."**

   You can insert data into only one table at a time. In this case, you've selected the Products table because it holds all the information for each item at the store. After information is added to the database, Dreamweaver redirects the visitor to another page. You'll set this up next.

10. **Click the Browse button. In the site's root folder, select the file *index.php*.**

    After adding a new product to the database, your staff is taken to the Products page (the one you created in the previous chapter). Since the newly added product is part of the database, browsing the products catalog reveals the newly added item.

Tip: You could also choose *add.php* for Step 10 if you wanted to quickly add multiple product records. Then, after one product was inserted into the database, the administrator would return to the product insertion form again, ready to input the next product.

11. **In the "Form fields" list, select "productID." Click the Remove (minus sign [–]) button to remove this field.**

    In some cases, the database itself fills in certain fields. For instance, every product in the database has its own unique ID—the table's primary key, which is generated by the database. When a new record is added, the database creates a new, unique number and stores it in the productID column. Since you don't want anyone entering the wrong information here, you should remove it from the form Dreamweaver is about to create.

12. **Select the productName column. In the Label field, change the label to "Product Name:".**

    The label you type here doesn't affect your database in any way. It's just the text that visitors see next to the form field. (You'll do the same thing with each field, to make the labels reader-friendly.)

    You don't need to change any of the other options such as "Display as" or "Submit as." You'll often change the "Display as" option, which changes what type of form element—like a checkbox or menu—Dreamweaver displays (you'll see this in the next step). However, you'll probably never change the "Submit as" option, which determines how Dreamweaver submits the data to the database. Dreamweaver figures this out correctly based on the design of your database.

13. **Select the "description" column. From the "Display as" menu, choose "Text area".**

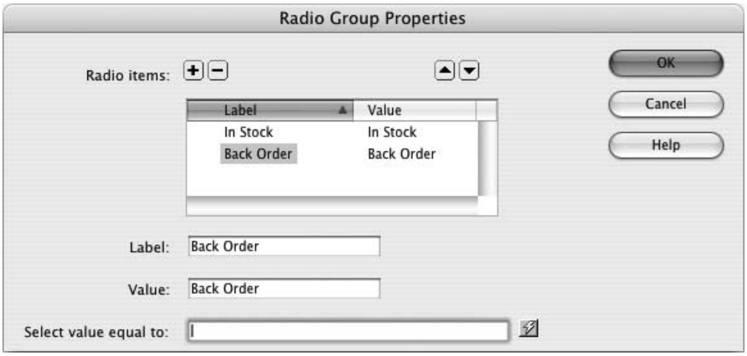
    The label for this column is fine, but since a description may be anywhere from several sentences to several paragraphs, it's a good idea to provide a large text box for collecting the product description. A regular text box is just one line, so changing the "Display as" option to a "Text area" (a multiline text box) provides plenty of space to accept visitor input (see page 408 for more information on multiline text boxes).

14. **Select the inventory column, and then, in the Label field, change the label to "Inventory Status:".**

    The database tracks a product's inventory status: whether the product is in the warehouse or on back order with the manufacturer. You could let a store administrator type in the correct status, but that would take time, and, besides, he might make a mistake. (It wouldn't do for shoppers to see that the "Kudzu seeds" are on "Gack Order.") So you'll simplify the process by adding radio buttons.

15. **From the Display As menu, choose Radio Group, and then click the Radio Group Properties button.**

    The Radio Group Properties window appears (Figure 24-17). Here you're going to add the radio buttons you want to appear on the form. Remember, the value of each button must match the data stored in the database (see page 858).



Label	Value
In Stock	In Stock
Back Order	Back Order

Label: Back Order

Value: Back Order

Select value equal to:

Figure 24-17: *Use the Radio Group Properties window to add radio buttons to a form. Radio buttons make data entry faster and less error-prone.*

16. **In the Label field, replace "button1" with In Stock. Type *In Stock* in the Value field as well.**

    The label is what appears on the page, while the value is the information that gets stored in the database. You need to add one more button.

17. **Click the + button to add another radio button; repeat step 15, but type Back Order for the label and value of the second button.**

    The window should look like Figure 24-17.

18. **Click OK to close the Radio Group Properties window.**

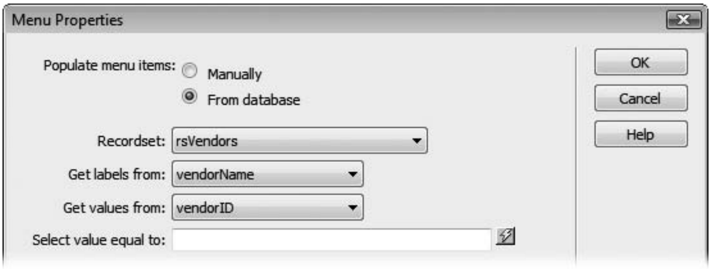
    Again, in an effort to speed up data entry and make sure the form is filled out correctly, the next two fields will be pull-down menus. First, you'll create a dynamic menu to display the list of vendors, as follows.

19. **Select the vendorID column, and then change its label to "Vendor:".**

    This column only stores a number; the vendor's name and contact information are stored in a different table. To make entering this information easier, you'll make a dynamic menu that lists all of the vendors' names. When somebody chooses a name from the menu, the appropriate *vendorID* number is submitted to the database.

20. **From the "Display as" menu, choose Menu. Then click the Menu Properties button.**

    The Menu Properties window opens (see Figure 24-18). Use this window to build the menu.



Menu Properties

Populate menu items: ○ Manually ● From database

Recordset: rsVendors

Get labels from: vendorName

Get values from: vendorID

Select value equal to:

Figure 24-18: *Dreamweaver can create dynamic pull-down menus (also known as pop-up menus) that get their labels and values from a database.*

21. **Click the "From database" radio button. Make sure that in the Recordset menu, "rsVendors" is selected.**

    You're telling Dreamweaver that the items to be listed in the menu are actually coming from a database query. In fact, they come from the recordset you created at the beginning of this tutorial—rsVendors.

22. **From the "Get labels from" menu, choose "vendorName." Then, from the "Get values from" menu, choose "vendorID."**

    The labels—the text that appears in the menu—are the names of each vendor. The value that's submitted with the form, meanwhile, is the vendor's ID number.

    You can skip the "Select value equal to" field. It's useful if you want a particular value to be preselected when the form loads, which is usually the case when you're *updating* a record in the database, since you need to display the current database information in order to update it.

23. **Click OK to close the window.**

    The product category is another instance where a pull-down menu makes sense. You'll follow the same procedure to add a pop-up menu listing the names of all the categories available at the store.

24. **Select the categoryID column, and then change its label to "Category:".**

    The next few steps should feel familiar.

25. **Repeat steps 20–23; use the "rsCategories" recordset, retrieve the label from the "categoryName" field, and set the value to "categoryID."**

    For the final field, you'll change the label and manually enter a default value.

26. **Select the image column, and then change its label to "Image File:".**

    Because not every product has an image, you'll change the default value to point to a graphic that's already been created—one used to indicate that no graphic is available.

27. **In the "Default value" box, type *none.gif*.**

    Finally, each product can be marked as "On Sale" or not.

28. **Select the onSale category, and then change the label to "On Sale:".**

    Either the product is on sale or it isn't. This kind of yes or no option is best represented by radio buttons.

29. **From the "Display as" menu, choose Radio Group, and then click the Radio Group Properties button.**

    Numbers in the database represent a product's sale status: If the onSale field has a value of 1, the product is on sale; if the value is 0, the product isn't on sale. Because 1 and 0 might not make sense to anyone using this Web page to add a product to the database, you'll use plain language words as labels.

30. **In the Label field, replace "button1" with *Yes*. Type *1* in the Value field as well.**

    Just one more button to add.

31. **Click the + button to add another radio button; repeat step 30, but type *No* for the label and *0* for the value of the second button. Click OK to close the radio button window.**

    At this point, the Record Insertion Form window should resemble Figure 24-16.

Note: Dreamweaver doesn't provide any way to return to the Insert Record Form wizard. If you accidentally close the window and insert the form before completing each of the steps in this section of the tutorial, you have to finish the form manually. Unless you're really sure how to do that, your best bet is to delete the form from the page, remove all server behaviors except for rsVendors and rsCategories, and then start at step 8 on page 869. Practice makes perfect, right?

32. **Click OK again to insert the form.**

    Dreamweaver adds a table, a form, and all the programming code necessary to add a new product to the database. The form has a blue background—one of Dreamweaver's signals that this form is special—and indicates that this form uses one of Dreamweaver's server behaviors (see Figure 23-26 on page 824 for information on how to hide or change this color).

33. **Choose File → Save.**

    You're nearly finished. You just have to finish up the design and take it for a test drive.

Finishing the Insert Form

To make your form ready for prime time, you'll spruce up its appearance and test it:

1. **Select the table containing the form fields.**

    The fastest method is to click anywhere inside the table and then, in the Tag selector, click the <table> tag (the one farthest to the right in the Tag selector). For other table selection techniques, see page 252.

2. **In the Property inspector, from the Align pop-up menu, choose Default.**

   The Default option aligns the table to the left without adding bandwidth-hogging HTML code. In addition, a little extra space is added around and inside each cell in a table. You'll remove that next.

3. **With the table still selected, in the Property inspector's CellPad, CellSpace, and Border boxes, type *0*.**

   You could repeat this step with the tables used for the two sets of radio buttons. In addition, if you don't like the way the buttons appear one on top of the other, you can move them. At this point, the labels, form fields, and tables in the form are fully editable. You could, for example, remove the labels and radio buttons from the table, place them side by side, and delete the small table.

   Now you'll apply a style to the table to improve its appearance.

4. **In the Property inspector, from the Class menu, choose insertForm.**

   The text inside the table should now be formatted to better match the site's style.

5. **Select each of the table cells in the left-hand column, and then, in the Property inspector, click the "Header" box.**

   You can select the cells by clicking the top cell, and then dragging down until all the cells in the left column are selected; Ctrl-clicking (⌘-clicking) each cell works as well. For other methods for selecting table cells, see page 253.

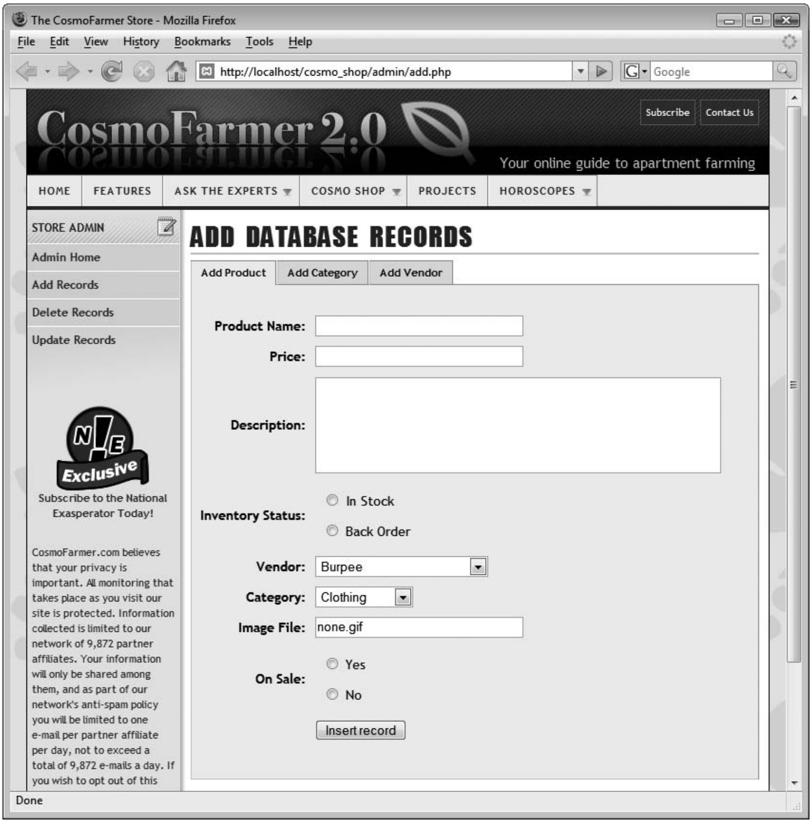
   The finished page should resemble Figure 24-19. Now you're ready to take the page for a spin.

6. **Press F12 (Ctrl-F12) to preview the page in a Web browser. Type information into each of the fields, and then click the "Insert record" button.**

   If you filled out all the fields correctly, you should see the product page you built in the last chapter. Click the category name of the new product you just added, or navigate through the product pages until you find the newly added item.

Note: If, when you submit the form, you get a page full of errors, you may be attempting to preview the page using a temporary file. See the Note on page 150 for an explanation.

You can enhance this page in many ways. For example, you can make sure that no one at CosmoFarmer accidentally inserts a new product without a price, a description, or any of the other required pieces of information; just add Dreamweaver CS3's new Spry Validation tools discussed on page 421. In addition, you could use the Insert Record Form wizard to add insert forms in the Add Category and Add Vendors tabbed panels. (See page 458 for information on how to work with Spry tabbed Panels.)



CosmoFarmer.com believes that your privacy is important. All monitoring that takes place as you visit our site is protected. Information collected is limited to our network of 9,872 partner affiliates. Your information will only be shared among them, and as part of our network's anti-spam policy you will be limited to one e-mail per partner affiliate per day, not to exceed a total of 9,872 e-mails a day. If you wish to opt out of this

Product Name:	
Price:	
Description:	
Inventory Status:	○ In Stock ○ Back Order
Vendor:	Burpee
Category:	Clothing
Image File:	none.gif
On Sale:	○ Yes ○ No

Figure 24-19: *No database-driven site would be complete without a way to add new records to the database. Use forms like this one to collect newsletter sign-up information, collect order and payment details, or just create an online guest book.*

Building a Page for Editing Database Records

If employees at CosmoFarmer type the wrong information for a particular product and have no way to correct it, they could be in a lot of trouble. After all, they'd be losing money hand over fist if the site were selling those $598 CAT Indoor Lawn Tractors for only $5.98. That said, here's how to add an update-record page to the site.

Linking to the update page

An update page is very much like an insert-record page; the only difference is that the form is already filled out with information about a particular record. First, you have to tell the update page which product it's supposed to update. To do so, you must add a link to the product-details page you built in the last chapter.

1. **In the local site's root folder, open the file named *product.php*.**

   This page lists details for a particular product. As you may recall from last chapter, this page is itself accessed from the *index.php* page, which displays a listing of all products in the database. By clicking the name of a product on that page, the *product.php* page retrieves and displays information on just that product.

Now you need to create a link on this page that, when clicked, takes a visitor to an update page for the particular product.

2. **Click to the right of the Inventory Status line, and then hit Enter (Return) to create a new, blank paragraph. Type *Edit This Information*. Select the text you just typed, and then, in the Property inspector, next to the Link field, click the "Browse for File" button (the little folder icon).**

   The standard Open File dialog box appears. (If you installed the extension described on page 818, you could also use the Go To Detail Page server behavior.)

3. **In the admin folder, navigate to and select the file called *update.php*, but don't close the window yet.**

   You need to add some additional information, which identifies the product that needs updating, to the end of the URL.

4. **Click the Parameters button to open the Parameters window. Click the name column, and then type productID.**

   The Parameters button lets you add a URL parameter to the end of a link, letting you pass information on to another page. In this case, you're passing on a dynamic piece of data—the product ID number for the item currently displayed on the Product Details page.

5. **Press Tab twice to hop to the Value column. Click the dynamic data button (the lightning bolt).**

   The Dynamic Data window opens. Here you can select data that you've already added to the Bindings panel, such as columns from a recordset.

6. **From the rsDetails recordset, select the item "productID," and then click OK.**

   (You may need to click the + button to the left of the word Recordset to see this option.) The link is nearly complete.

7. **Click OK to close the Parameters window. Click OK once again to close the Select File window and apply the link.**

   When you're all done, the Property inspector's link box should look like this:

   ```
   admin/edit.php?productID=<?php echo $row_rsDetails['productID']; ?>
   ```

8. **Choose File → Save.**

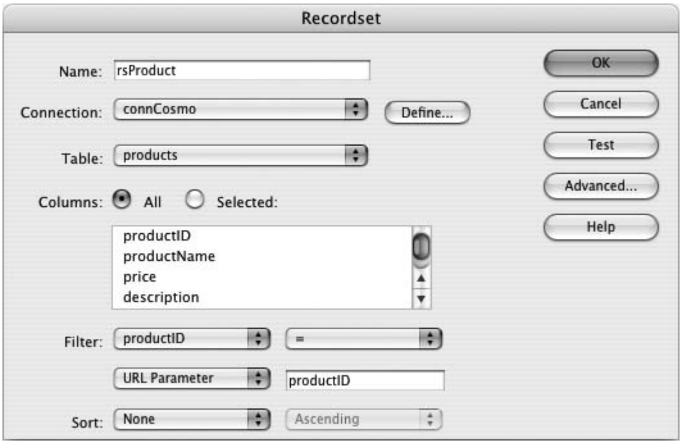
Note: You probably wouldn't want a link like this to appear for the average visitor to your site. After all, customers shouldn't be changing information on the products you sell. In the next chapter on page 917, you'll learn how to hide this link from unauthorized eyes.

Creating the update page

Now that the initial legwork is out of the way, you're ready to build the actual Record Update Form. To start, you'll add a filtered recordset to retrieve information for the product to be updated:

1. **In the admin folder, open the file *edit.php*.**
2. **Add a recordset, using any of the methods described on page 782. For example, choose Insert → Data Objects → Recordset.**

   The Recordset window opens. Make sure the simple options are displayed, as shown in Figure 24-20.



Recordset

Name: rsProduct

Connection: connCosmo

Table: products

Columns: ● All ○ Selected:

productID
productName
price
description

Filter: productID =

URL Parameter productID

Sort: None Ascending

Figure 24-20: *When you filter on a table's primary key (productID, in this case) using the = operator, the recordset never retrieves more than one record.*

3. **In the Name field, type *rsProduct*; from the Connection menu, choose connCosmo, and from the Tables menu, select Products. Leave the All button selected.**

   Next, add a filter to the recordset. This filter ensures that the recordset retrieves only a single record—the product you wish to update.

4. **From the Filter menu, select productID. From the Comparison menu, select =. From the Source menu, choose URL Parameter. Finally, make sure the last field in the Filter area of the window says productID.**

   After you selected productID, Dreamweaver most likely filled in the other three options for you. When creating a filtered recordset, Dreamweaver assumes you'll use a URL parameter with the same name as the selected field. The Recordset window should now look like Figure 24-20. In essence, it instructs the recordset to retrieve only the record whose productID matches the number passed in the URL parameter named productID (that's the name you supplied as part of the link in step 4 on page 876).

5. **Click OK to close the window and add the recordset to the page.**

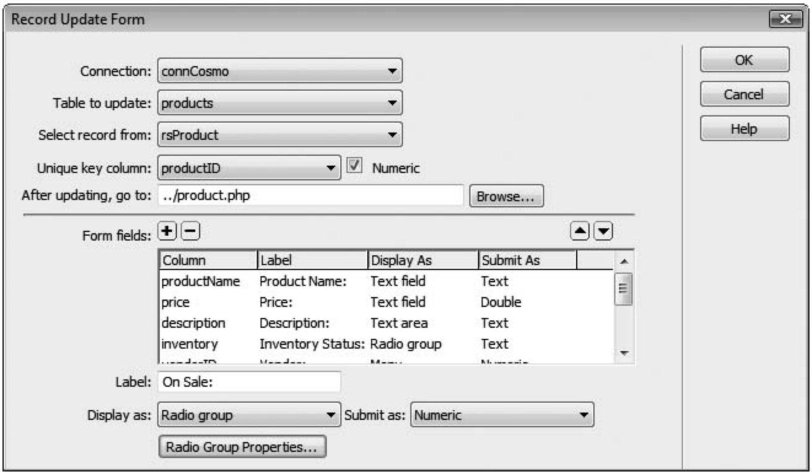
   Next, you'll create two more recordsets—a listing of all vendors and a listing of product categories. You'll use them to create dynamic menus, just as you did on the insert form.

6. **Follow steps 2–7 from the "Adding an Insert Product Page" part of this tutorial (see page 868) to create new rsVendors and rsCategories recordsets.**

   (You can also copy those recordsets from the insert product page as described on page 798). The hard part's behind you. You can now use Dreamweaver's Update Record Form tool to finish the page.

7. **Click directly underneath the green line of the Edit Record headline. Choose Insert → Data Objects → Update Record → Record Update Form Wizard.**

   The Record Update Form window opens (see Figure 24-21). Next, you'll specify the recordset and fields the form should update.



Record Update Form

Connection: connCosmo
Table to update: products
Select record from: rsProduct
Unique key column: productID ☑ Numeric
After updating, go to: ../product.php

Form fields:

Column	Label	Display As	Submit As
productName	Product Name:	Text field	Text
price	Price:	Text field	Double
description	Description:	Text area	Text
inventory	Inventory Status:	Radio group	Text
vendorID	Vendor:	Menu	Numeric

Label: On Sale:
Display as: Radio group
Submit as: Numeric

Figure 24-21: *Dreamweaver's Record Update Form wizard makes very quick work of creating pages to update records in a database.*

8. **From the Connection menu, select connCosmo. Make these selections for the next three menus: "products" in the "Table to update" menu, "rsProduct" in the "Select record from" menu, and "productID" in the "Unique key column."**

   Next, you need to specify which page appears after someone updates the record. Since the update page lets you edit a single product, it makes sense that after submitting any changes, you should see the newly updated information on that product's detail page.

Note: As with the Insert Record Form wizard, once you close the Update Record Form wizard window, there's no way to return to it (see the Note on page 873).

9. **Click the Browse button. In the Select File window, navigate to and select the file *product.php*. Click OK to choose the file.**

   Now you must specify which fields appear in the form. You also need to change which type of form element they should use, and edit their labels. This process is very similar to the Insert Record form; it's summarized in the following steps.

10. **In the "Form field" list, select "productID"; click the Remove (minus sign [–]) button to remove this field from the list.**

    Since productID is a primary key generated by the database, no one should be allowed to change it.

    Next, you'll change the text label that appears next to a couple of the fields.

11. **Select the "productName" form field, and then change its label to "Product Name."**

    Next, you'll provide some more room for lengthy descriptions of each product.

12. **Select the "description" column. From the Display As menu, choose Text Area.**

    As with the insert product page, inventory status information is better displayed with a simple pair of radio buttons. You'll add those now.

13. **Select the "inventory" column. Change the label to "Inventory Status:", from the Display As menu, choose Radio Group.**

    You now need to give Dreamweaver a bit of information about the radio buttons you wish to add to the page.

14. **Click the Radio Group Properties button.**

    The Radio Group Properties window appears (Figure 24-17). You need to add the radio buttons you want to appear on the form. The value of each button must match the data stored in the database.

15. **In the Label field, replace *button1* with *In Stock*. Type *In Stock* into the Value field, too.**

    The label appears on the page, while the value is stored in the database. You need to add one more button.

16. **Click the + button to add another radio button; repeat step 15, but type *Back Order* for the label and value of the second button.**

    The window should look like Figure 24-17, except that the "Select value equal to" box is filled with the programming code necessary to select the correct button. Since this is an update form, one of the buttons should *already* be selected when the page loads—information stored in the recordset determines which button is selected.

17. **Click OK to close the Radio Group Properties window.**

    Again, in an effort to speed up data entry and make sure the form is filled out correctly, the next two fields will be pull-down menus. First, you'll create a dynamic menu to display the list of vendors.

18. **Select the vendorID column, and then change the label to "Vendor:". From the Display As menu, choose Menu; click the Menu Properties button.**

    The Menu Properties window opens (see Figure 24-18).

19. **Click the "From database" radio button, make sure "rsVendors" is selected in the Recordset menu, and then, from the "Get labels from" menu, choose "vendorName." Now, from the "Get values from" menu, choose "vendorID."**

    Leave the "Select value equal to" field as is. Dreamweaver automatically selects the appropriate choice, based on which vendor manufactures the product.

20. **Click OK to close the Menu Properties window.**

    You need to repeat the process for the product categories menu.

21. **Select the "categoryID" column, and then change the label to "Category:". From the Display As menu, choose Menu; click the Menu Properties button.**

    The Menu Properties window opens.

22. **Click the "From database" radio button, make sure "rsCategories" is selected in the Recordset menu, and then choose "categoryName" from the "Get labels from" menu. Now, from the "Get values from" menu, choose "categoryID." Click OK to close the Menu Properties window.**

    As with the previous menu, Dreamweaver automatically adds the correct code to make sure the category for the product being edited is correctly selected when this update page loads.

23. **Select the image form field, and then change its label to "Image File:".**

    Finally, you'll provide a way to indicate whether a product is on sale.

24. **Select the "onSale" category, and then change the label to "On Sale:".**

    Either the product is on sale or it isn't. You can best represent this kind of yes or no option by radio buttons.

25. **From the Display As menu, choose Radio Group, and click the Radio Group Properties button.**

    Numbers in the database represent the sale status of a product: if the onSale field has a value of 1, the product is on sale; if the value is 0, the product isn't on sale. Because 1 and 0 might not make sense to anyone using this Web page to add a product to the database, you'll use plain language words as labels.

26. **In the Label field, replace *button1* with *Yes*. Type *1* in the Value field as well.**

    Just one more button to add.

27. **Click the + button to add another radio button; repeat step 30, but type *No* for the label and *0* for the value of the second button. Click OK to close the radio button window.**

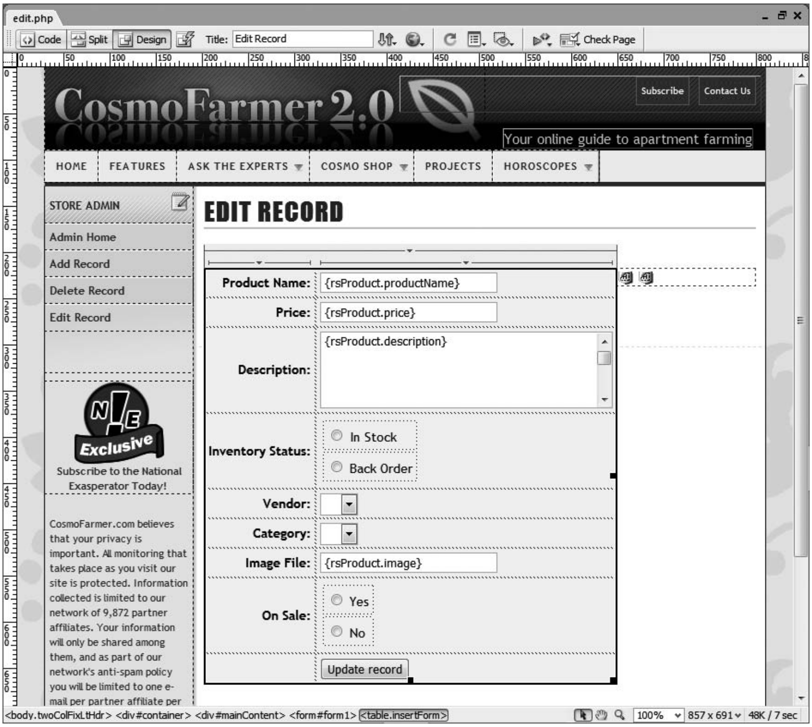
    At this point, the Record Update Form window should resemble Figure 24-21.

28. **Click OK to close the Record Update Form window.**

    Dreamweaver inserts a table, form, form fields, and programming code to the update page. All that's left are some final cosmetic touches.

29. **Repeat steps 1–5 from the "Finishing the Insert Form" part of this tutorial (page 873).**

    Doing so properly formats the form and adds the necessary form validation behavior. Your finished page should resemble Figure 24-22.



CosmoFarmer.com believes that your privacy is important. All monitoring that takes place as you visit our site is protected. Information collected is limited to our network of 9,872 partner affiliates. Your information will only be shared among them, and as part of our network's anti-spam policy you will be limited to one e-mail per partner affiliate per

Product Name:	{rsProduct.productName}
Price:	{rsProduct.price}
Description:	{rsProduct.description}
Inventory Status:	○ In Stock ○ Back Order
Vendor:	
Category:	
Image File:	{rsProduct.image}
On Sale:	○ Yes ○ No

<body.twoColFixLtHdr> <div#container> <div#mainContent> <form#form1> <table.insertForm>

Figure 24-22: *When you're working in Design view, Dreamweaver highlights dynamic areas of the page–like this update form–in light blue. In Live Data view (see page 812), the same areas change to yellow. If you'd like to hide this coloring, open the Preferences window (Edit → Preferences [Dreamweaver → Preferences]), select the Highlighting category, and then turn off the two Live Data checkboxes.*

30. **Save this page and close it.**

    To get a feel for what you've done, it's time to test your application.

31. **In the root folder (the *cosmo_shop* folder), open the *index.php* page. Press F12 (Option-F12) to preview it in a browser.**

    The page lists the products in the database. Take a close look at one product in particular.

32. **Click the name of any product in the list.**

    A details page for that product appears.

33. **Click the Edit This Information link near the bottom.**

    The Update Product page appears, with the form already completed.

34. **Change some of the information on the form, and then click the "Update record" button.**

    Voilà! You're taken back to the details page for this product listing, which proudly displays the freshly edited content.

Creating and Linking to the Delete Page

Obviously, if a vendor stops manufacturing a product, or the staff at CosmoFarmer decides to discontinue an item, you need a way to remove a product listing from the database.

Adding a link on the details page

To begin, you must provide a link to delete the product. A good place for this would be on the details page of each product. Since you've already added an Edit This Information link to this page, you must now add a Delete This Product link:

1. **Open the file *product.php*.**

   Add a link that leads to a delete page.

2. **Near the bottom of the page, click to the right of the text you added earlier: Edit This Information. Press the Space bar, followed by the | character and another space; type *Delete This Product*.**

   Now you'll link this phrase to the delete page.

Note: It's easy to accidentally click into the "Edit This Information" link. Doing so will make any text you type a part of that link. If you're in this situation, just delete the new text you typed, click the <a> in the Tag selector at the bottom of the document window, and then press the right arrow key. This moves the insertion point to the right of the link.

3. **Select the text "Delete This Product," and then, in the Property inspector, click the "Browse for File" button (the little folder icon). In the admin folder, navigate to and select the file *delete.php*, but don't close the window yet.**

   You need to add the information that lets the delete page form know which product it should delete.

4. **Follow steps 4–7 in the "Linking to the update page" part of this tutorial (page 876).**

   Doing so creates a link that not only goes to the delete page, but also passes along the ID number of the product to be deleted.

5. **Save and close this page.**

You'll probably find yourself needing to use the same recordset on several pages on a site. For example, on the product detail page you created in the last tutorial, you created a filtered recordset which retrieved information on a single product (based on an ID number passed in the URL). You recreated that same recordset on the update record page you created on page 877. You'll also need it for the delete page (in order to select a single product to delete). Instead of recreating that recordset yet again, you'll just copy it from another page.

6. **Open the file *edit.php*.**

   You'll copy the recordset from the Bindings panel, so make sure it's open (Window → Bindings).

7. **Right-click (Option-click) on the rsProduct recordset, and then, from the pop-up menu, choose Copy.**

   You can also select the recordset in the Bindings panel, click the contextual menu that appears in the panels' top-right corner, and then select Copy. Either way, you've copied the programming code necessary for the delete page.

Creating the delete page

You've just created a link to the delete page; now you need to make the delete page do its stuff:

1. **Open the file *delete.php*.**

   This is where you'll paste the recordset that you copied a moment ago.

2. **Make sure the Bindings panel is open. Right-click (Control-click) in the empty area of the panel; from the shortcut menu, choose Paste.**

   Dreamweaver pastes all the programming code to create a recordset. This method is a fast way to reuse a recordset.

3. **In the Bindings panel, expand the recordset listing by clicking the small + button (arrow on Macs) to the left of the recordset.**

   Don't click the *large* + button, which lets you add additional recordsets. You just want to see an expanded listing of recordset columns so you can add some dynamic data to the page.

4. **Drag the productName column from the Bindings panel and drop it onto the document window, just to the right of the text "Product to delete."**

   This action inserts dynamic data into the page. When this page appears in a Web browser, the name of a product appears in bold type.

5. **Click the empty space just below the name of the product. Choose Insert → Form → Form.**

   A red, dotted line—the boundaries of the form—appears on the page. You need to set the form's *action* property (a URL to the page that collects the form information).

6. **In the Property inspector, click the action box, and then type *delete.php*.**

   Now when this form is submitted, its contents will be sent…to itself! This is a common maneuver with dynamic pages, which often do double duty depending on how you access them. In this case, when the form is submitted, the programming code (which you'll add in a minute) receives the request to delete a particular product. Instead of displaying the page and form again, it merely deletes the record from the database, and then redirects the browser to a different page (see page 864 for more detail on how this works).

7. **Choose Insert → Form → Button. (If the "Input Tag Accessibility Attributes" window appears, click Cancel to close it.) In the Property inspector, change the value to Delete.**

   Currently the button's label says Submit, but a more descriptive term like Delete would be better.

8. **In the Property inspector, change the Value box to read Delete.**

   This button, when clicked, removes one product from the database. However, you need to identify the product as well. A hidden form field containing the product's ID number will do the trick.

9. **In the document window, click to the right of the button you just added, and then choose Insert → Form → Hidden Field.**

   When you view the Web page, you don't see a hidden field, but it provides useful information when the form is submitted. In this case, it needs to supply the product ID.

10. **With the hidden field selected, change its name in the Property inspector from "hiddenField" to "productID".**

    This step and the next are similar to adding a parameter to the end of a URL (for example, to link from a list of records to a detail page for a single record). The only difference is that instead of placing the product's ID number in a URL, it's now embedded within a form. Now you need to add the product ID that's retrieved from the database.

11. **In the Property inspector click, the lightning bolt to the right of the value box.**

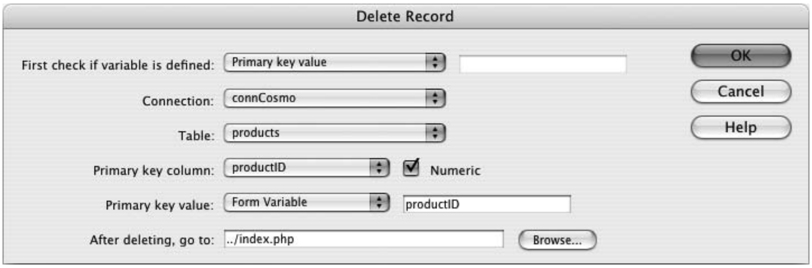
    The Dynamic Data window opens, displaying the rsProduct recordset.

12. **From the list of fields, select "productID", and then click OK to close the Dynamic Data window.**

    Now when someone views this page, the doomed product's ID number is stored in this hidden field.

13. **Open the Server Behaviors panel (Window → Server Behaviors). Click the + button, and then select Delete Record.**

    The Delete Record window appears (see Figure 24-23).



Delete Record

First check if variable is defined: Primary key value

Connection: connCosmo

Table: products

Primary key column: productID ☑ Numeric

Primary key value: Form Variable productID

After deleting, go to: ../index.php

***Figure 24-23:** The Delete Record behavior adds all the necessary programming code to remove a record from the database. All you need to make it work is a recordset that retrieves a single record—and a form with a Delete button and a hidden field containing the record's primary key value.*

14. **From the first menu, select Primary Key Value.**

    This step tells the server behavior that it shouldn't delete the record until it's given a primary key value (you define when this happens in steps 17 and 18 below).

15. **From the Connection pop-up menu, choose connCosmo.**

    Now tell Dreamweaver which table the record belongs to.

16. **From the "Table" menu, choose "products."**

    This menu indicates the table containing the record that is to be deleted. You next have to specify the primary key (see page 769) of the record to delete.

17. **From the "Primary Key Column" menu, select "productID," and make sure the Numeric box is checked.**

    Now you need to let the server behavior know where the primary key value will be coming from. In this case, the ID number for the product to be deleted is embedded in a hidden form field named productID.

18. **From the Primary Key Value field, choose Form Variable, and make sure *productID* appears in the box to the right.**

    To finish filling out this window, you'll just tell Dreamweaver which page should appear after someone deletes the record.

19. **In the Property inspector, click the "Browse for File" button. In the root folder (cosmo_store), navigate to and select the file *index.php*.**

    The Delete Record window should now look like Figure 24-23.

20. **Click OK.**

    Dreamweaver adds the Delete Record server behavior to the page. You've done it! Now you need to test it out.

21. **Save and close this page. Open the *index.php* page. Press the F12 key (Option-F12) to preview it in a browser.**

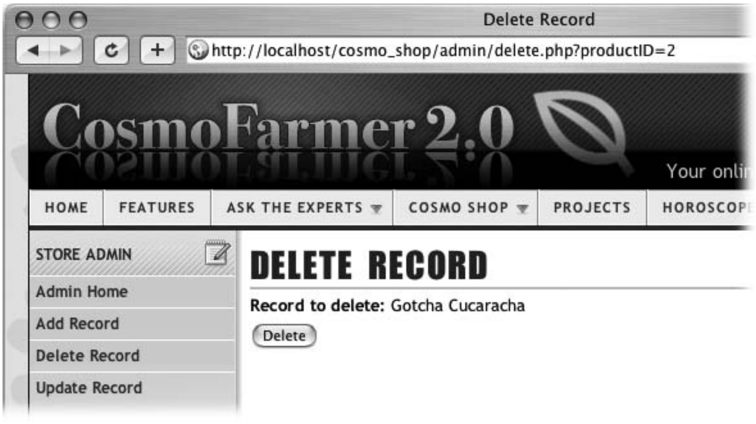
    The page lists the products in the database. Take a closer look now at a specific item.

22. **Click the name of any product in the list.**

    A details page for that product appears.

23. **Click the Delete This Product link near the bottom.**

    The Delete Product page appears (see Figure 24-24). Notice that both the product name and a Delete button appear.



Record to delete: Gotcha Cucaracha

Figure 24-24:
When you first access the page (from a link on a product details page), it displays the confirmation shown here. But when the Delete button is clicked, the page is reloaded, and a Delete command is sent to the database.

24. **Click the Delete button to remove the item.**

    Don't worry, you can always insert more products later!

    In any case, you'll note that that the product is no longer listed in the Product listings.

Of course, in the real world, you wouldn't want just anybody deleting, adding, or editing products on an e-commerce Web site. So in the next chapter, you'll learn how to keep prying eyes and mischievous fingers away from your coveted insert, update, and delete pages.

CHAPTER

25

Advanced Dynamic Site Features

Dreamweaver's basic database capabilities are impressive. But there may come a time when you need to dig deeper into the program to build successful Web applications. Dreamweaver's advanced features let you, the mere mortal, do things that the pros do every day, like password-protect pages; display (or hide) content based on database results; and access information from forms, cookies, and URLs.

Password-Protecting Web Pages

Although Dreamweaver lets you create Web pages that can add, edit, and delete records from a database, your e-business wouldn't last very long if just *anyone* could remove orders from your online ordering system or view credit card information stored in your customers' records. And certainly your company's executives wouldn't be happy if someone accessed the staff directory database and changed the boss's title from, for example, CEO to Chief Bozo. For these and other reasons, Dreamweaver provides a simple set of tools for locking your pages away from prying eyes.

The User Authentication server behaviors can password-protect any page on your site. With this feature, you can limit areas of your site to registered users only, allow customers to access and update their contact information, create maintenance pages accessible only to administrators, or personalize Web pages with customized messages ("Welcome back, Dave").

Note for ASP.NET Users: Dreamweaver doesn't provide any User Authentication behaviors for ASP.NET. If you want to add these to your ASP.NET pages, you'll have to program them yourself or try a commercial extension like ASP.NET Authentication Suite from WebXcel (*www.webxel-dw.co.uk*).

To password-protect pages on your site, you'll need to get several elements in order:

- A database table containing visitors' login information.
- A registration form for adding new visitors. (This is an optional step, but it's frequently useful when you want to automate the process of adding user login information to the database.)
- A login form.
- One or more pages that need to be password-protected.

The Users Table

To password protect your Web pages, your database must hold several pieces of information about the people who can access those secret pages. For example, each visitor must have a user name and password to type in when he attempts to log into your site. If the name and password match a record in the database, then he's logged into the site and can access password-protected pages.

You might also need to include a field in the record for assigning an *access level* to each person. This way, your site can have multiple sections, accessible by different groups of people. Dreamweaver provides tools not only to require a proper name and password, but also to allow access to only those with the proper clearance level.

For example, if your site has a members-only section that affords registered visitors extra content or special features, you could assign the level of "member" to everyone who registers and give them access to these pages. However, you want only your site's administrators and staff to be able to update a product database or retrieve sales records, so you would give these users a level of "administrator" for access to these areas.

At a minimum, then, your database needs a users table with three fields (user name, password, and access level). You can either use a standalone table or incorporate this information into another table. For example, if you require people to provide their names, addresses, email addresses, and so on when registering with your site, you could include the three login fields in this table. For an e-commerce system, login information could be stored in the table holding customer information.

Tip: Most database systems let you assign a default value to a column. That way, when someone creates a new record and supplies no information for the column, the application enters the default value instead.

For starters, it's a good idea to assign a default value for the access-level field. You can set your database to assign the lowest access level—"guest," say—whenever someone creates a new user record. In this way, if you use a Web form for collecting and creating a new member, you can omit a form field for assigning an access level. This method is a good security precaution, as adept (and malicious) Web surfers could submit a fake form with a higher access level, potentially granting them access to sensitive areas of your site.

Creating a Registration Form

Once you've added a users table to your database, you'll need a way to add new members. If you plan to use password protection for sensitive pages that only your site's staff should access, you probably *shouldn't* create a Web form for adding new administrative members. You'd run the risk of someone finding the form and adding herself to the list of administrators. In such cases, you're better off adding the proper login records in the database system itself—using phpMyAdmin, Microsoft Access, SQL Server, or MySQL Monitor, for example.

Note: If you do create a Web form for adding new members with a high access level, password-protect this form! Otherwise, anyone stumbling upon it could add new administrative members–and from there, Pandora's box would be open.

On the other hand, if you want to let *lots* of people sign up as members of your site, you might want to add a registration form that adds them to the list of the site's members *automatically*. This setup would free you from the headache of manually assigning user names and passwords for everyone who wants to become a member.

If the site already includes a form for collecting visitor information, you can simply add the proper user fields to this form. Say your site includes a "Sign up for our email newsletter" page that collects a visitor's name, email address, and other contact information. You could add a field called *user name* and another called *password*.

Tip: It's common to use an email address as a person's user name for password-protected pages. If you're already collecting an email address, exclude the user name field from the form.

When the visitor submits the form, the Web application adds all of these fields to the database. (To add records to a database using a Web form, see page 843.) While the process of creating a new member for password-protected pages is basically the same as adding a new record to a database, there's one additional step: you must make sure that every visitor has a unique user name.

Dreamweaver's Check New User Name server behavior ensures that each user name submitted in the form is unique. If the name already exists, the server won't add the new record to the database and will redirect the visitor to another page. To apply this server behavior, follow these steps:

1. **Add an insert-record form to a dynamic page.**

   The form should include fields for a user name and password. You might also add a field for an access level, if that's how you've structured your site. However, for a form that's accessible to the public, it's best to use the database to set a default value for this; see the Tip on the opposite page. (You'll need to use Dreamweaver's Insert Record server behavior. Creating insert-record forms is described on page 844.)

2. **Make sure the Server Behaviors panel is open (Window → Server Behaviors). Click the Add (+) button and, from the pop-up menu, choose User Authentication → Check New Username.**

   You can also use the User Authentication menu on the Application tab of the Insert Bar (see Figure 25-1). Either way, the Check New Username window appears (see Figure 25-2).

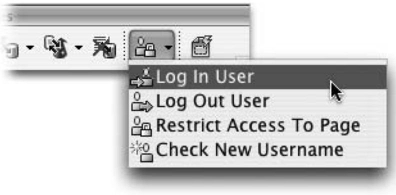
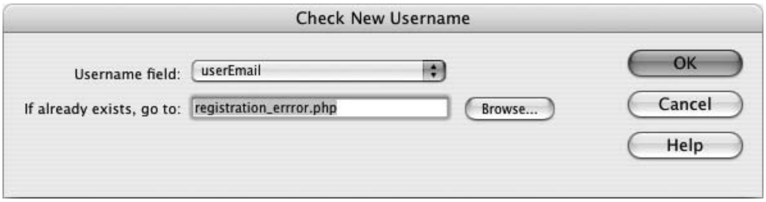


Figure 25-1:
Dreamweaver provides access to all user-authentication server behaviors from the Application tab of the Insert bar.


Check New Username

Username field: userEmail

If already exists, go to: registration_errror.php

Figure 25-2:
When adding a new person to your database, the Check New Username server behavior lets you verify that the user name isn't already in use by another person.

3. **Select the name of the database field that stores each user name.**

   Note that this is the name of the column in the *database*, not the name of the form field on the Web page. This field doesn't necessarily need to be named "userName." It could be "login" or something else.

4. **Click Browse and select a Web page.**

   Here, choose the page that will open if the user name is already assigned to someone else. This page (which you should create before applying this behavior) should include a note to your visitor, clearly spelling out the problem (the user name just supplied is already in use and therefore unavailable). To make re-entering information easier for your guest, you should include the insert form on this page as well, or provide a link back to the registration-form page.

5. **Click OK to close the window and add the server behavior to the page.**

   Now when someone fills out the registration form, this behavior will kick in and make sure that no one else has the same user name.

Note: Registering a new member doesn't automatically log him into the site. He'll still need to go to a login page (described next).

After inserting the server behavior, Dreamweaver lists it in the Server Behaviors panel. If you wish to change any of its properties, double-click its name in the panel to reopen the Check New Username window (Figure 25-2). To delete the behavior, select it in the Server Behaviors panel, and then click the Remove (minus sign [-]) button.

Creating the Login Page

To access a password-protected page, your visitor must first log into the site using a Web form. This simple Web form should contain just two fields—a user name field and a password field—and a Submit button.

When someone attempts to log in, the values she types into the form are compared with the user name and password columns in the database. If there's a match, then she's transported to another page—often the main page of a password-protected area of the site. If there is no matching record, then the visitor is carted away to a page of your creation—an "Access Denied!" page or maybe just the original login page.

To create a login page:

1. **Add a Web form to a dynamic Web page.**

   If your site includes password-protected pages aimed at a general audience of Web visitors, you could place this form on your home page. Or you could create a dedicated login page (remembering to provide links to this page throughout your site). However, if you're creating a login for administrators, you might want to put the login form out of the way, so that it isn't noticed by the average visitor.

   Either way, the form should contain only a user name field, a password field, and a single Submit button. Naming the fields "username" and "password" (rather than keeping Dreamweaver's factory-set field names) will help with step 3.

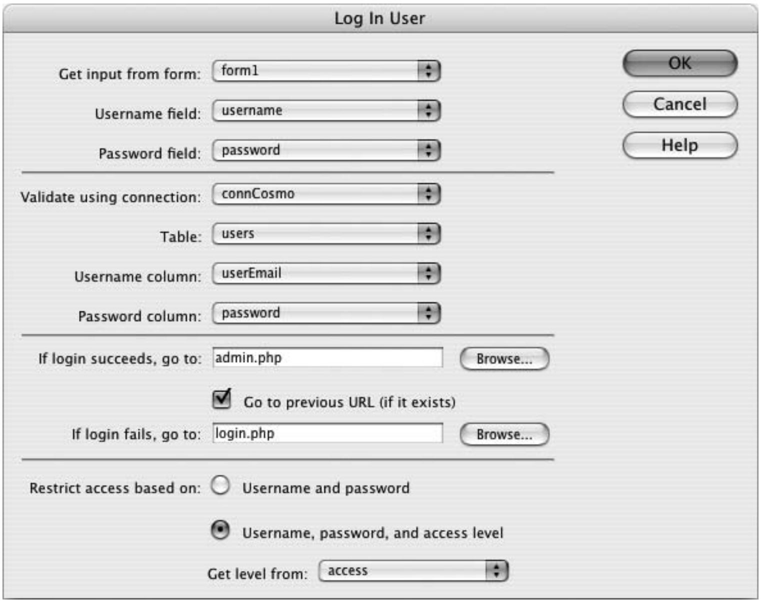
2. **Open the Server Behaviors panel (Window → Server Behaviors). Click the + button and choose User Authentication → Log In User.**

   You can also use the Data tab of the Insert bar (see Figure 25-1) or choose Insert → Data Objects → User Authentication → Log In User to open the Log In User window (see Figure 25-3).

3. **From the first three menus, select the names of the login form, the form field for collecting the user name, and the password field, respectively.**

   You're telling Dreamweaver which form (if the page has more than one) and which fields to use for comparison to the users table in the database.

Tip: Dreamweaver automatically makes these first three menu selections for you if the following things are in place: you've got just one form on the page; the first field on that form is the user name field; and the second field is the password field.



Log In User

Get input from form: form1

Username field: username

Password field: password

Validate using connection: connCosmo

Table: users

Username column: userEmail

Password column: password

If login succeeds, go to: admin.php

☑ Go to previous URL (if it exists)

If login fails, go to: login.php

Restrict access based on: ○ Username and password

● Username, password, and access level

Get level from: access

Figure 25-3: *Dreamweaver's Log In User server behavior gives visitors a way to log into your Web site, so they can visit password-protected pages. There are quite a few items to fill out here, but they're all straightforward. The last option lets you use access levels to limit pages of the site to particular groups of visitors—administrators, for example.*

4. **From the "Validate using connection" menu, choose the name of the database connection.**

   This should be the database that contains the table with user login information.

5. **From the Table menu, choose the name of the users table.**

   This is the table described on page 888, which includes the username, password, and access level for anyone attempting to log in.

6. **From the "Username column" menu, select the database column that stores names. From the "Password column" menu, choose the database column for passwords.**

   The User Authentication server behavior will search these two database columns for a record that matches the values your visitor types into the form.

7. **To the right of the "If login succeeds" field, click the Browse button; navigate to and select a page from your site.**

   Most of the time, this will be the main page for a password-protected area of the site. If the site contains a members-only section, then, after logging in, the visitor would arrive at the Members page. If you're adding features for administering the site—adding, deleting, and editing database info, for example—create a main Administrators page with links to all of the database administration pages.

8. **Turn on the "Go to previous URL" checkbox.**

This option is a little confusing, but very convenient. Imagine a visitor stumbling across a password-protected page (you'll learn how to protect pages in the next section). He simply comes across a link to a password-protected page and clicks it. Of course, since he hasn't logged in, he's denied access to the page and sent to another page. At this point, you're probably redirecting him to the login page, so he can log in and continue clicking his way through your site.

That's where this feature comes in handy. By turning on this box, you permit the login form to take the visitor *back* to the page he couldn't get past at the outset. In other words, the visitor tries to access a password-protected page (*any* password-protected page in the site); he's not logged in, so he's sent to the login page. After successfully logging in, he's taken directly to the page he first tried to access (*not* the page you specified in step 7). This is very convenient for visitors who bookmark password-protected pages in your site, since it saves them the hassle of having to log in and then navigate to the page they wanted in the first place!

Note for PHP Users: While the "Go to Previous URL" option is convenient, it won't remember URL parameters from the previous page (this limitation is an issue in the PHP/MySQL server model only). For example, say you try to visit a page used for updating a record in a database: *update_record.php?recordID=2.* If the page is password protected and you're not logged in, then you'll be sent to a login page. After successfully logging back in, you'll be returned to *update_record.php,* but the URL parameter *recordID=2* won't be available. In other words, returning to the previous URL won't end up giving you the page you wanted–for example, an edit form for record number 2.

9. **To the right of the "If login fails" field, click Browse; navigate to and select a page from your site.**

   This page, which you need to create in advance, should explain that the user name and password were not correct. Since the visitor may have just made a typo, it's polite to either include another login form on this page or a link back to the login page.

10. **If the database includes a column for storing an access level, select the "Username, password, and access level" radio button.**

    This option not only lets folks log into the site, but also tracks their access levels. In this way, you can limit areas of your site to people with the proper access level—administrators, for example.

11. **From the "Get level from" pop-up menu, select the name of the database column that contains visitors' access levels.**

    Dreamweaver lists all of the columns in the table you selected in step 5. If the table doesn't have a column for this information, go to your database application and add it, or deselect the Access Level radio button. (Even if you don't currently have plans for offering different levels of access, it's a good idea to keep this option in mind. In the future, you may very well want to add special pages for administrators or Super Premium Members. Without an access level, anyone who has a user name and password will be able to visit those pages.)

12. **Click OK to close the window and apply the behavior to the page.**

You can edit or delete this behavior by double-clicking its name in the Server Behaviors panel.

POWER USERS' CLINIC

Logging In: Behind the Scenes

The Log In User server behavior checks to see if the user name and password submitted by a form matches a user name and password in the database. If it does, the behavior generates two session variables (see page 903): MM_Username and MM_UserGroup for the PHP/MySQL server model and MM_Username and MM_UserAuthorization for the other server models. The first one (MM_Username) stores the user name of the logged-in visitor; the second (Mm_UserGroup or MM_UserAuthorization) stores the visitor's access level. (The MM stands for Macromedia, since these server behaviors were written before Adobe owned Dreamweaver.) The variables follow visitors from page to page of the site, until they log out, close the browser, or don't do anything on the Web site for at least 20 minutes.

The password-protection scripts use these session variables to allow or deny access to a page. But you can take advantage of these variables in other ways. You can add MM_Username to the Bindings panel (see page 897), for example. You can then add it to your pages, like other dynamic data, for customized pages: "Welcome back Kotter176@aol.com."

Furthermore, since each user name is unique–just like a primary key–you can use the session variable to filter records in a recordset (see page 785). You could use this technique, for instance, when a logged-in visitor wishes to see all of his contact information. Create a recordset that filters the user table by the session variable.

You can also use the MM_UserGroup (PHP) or MM_UserAuthorization variable to control the display of certain areas of a page. For example, while regular members of your Web site might see a simple listing of products on a dynamic catalog page, administrators might see additional items like "Edit this product" and "Delete this product" buttons. The tutorial at the end of this chapter has an example of this scheme in action (see page 917).

The Log Out User Behavior

Dreamweaver's Log *Out* User server behavior lets someone log out by clicking a link. Thereafter, her Web browser won't be able to load any password-protected pages in the site until she logs back in.

This setup is useful when a visitor shares her computer with others, maybe at the library or at school, because it provides a sense of security that she has the ability to log out. (It's not absolutely necessary, though; her computer destroys the cookie used to identify the session variable used to keep track of her login status, anyway, as soon as she quits her browser. Furthermore, if a certain amount of time passes without any activity—usually 20 minutes—the Web server automatically destroys the session variable, effectively logging out the visitor. Again, though, a logout link can be reassuring to your audience.)

To add a Log Out server behavior:

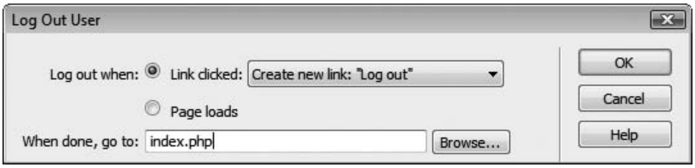
1. **Open a dynamic page.**

   Note that since this adds programming code to the page, it works only on dynamic pages. You can't add a logout link to a static HTML page. So if you want to provide this option on all pages of your site, you'll have to save each page in your site as a PHP, ASP, Cold Fusion, JSP, or other dynamic page that matches your server model.

2. **Click the page where you'd like to add a logout link.**

3. **Open the Server Behaviors panel (Window → Server Behaviors). Click the + button and, from the pop-up menu, choose User Authentication → Log Out User.**

   Alternatively you can use the Application tab of the Insert bar (see Figure 25-1) or choose Insert → Application Objects → User Authentication → Log Out User. In any case, the Log Out User window appears (see Figure 25-4).



Log Out User

Log out when: ● Link clicked: Create new link: "Log out"

○ Page loads

When done, go to: index.php

Figure 25-4: *To add a logout function to text or an image that's already on a page, simply select it and then apply the Log Out User server behavior.*

4. **Select one of the two radio buttons.**

   There are two ways a visitor can be logged out. You can log her out when a page loads or when she clicks a link. You'd use the first method when you want to automatically log someone out when she reaches a specific page. For example, say you create an online testing application, where students would sit at a computer and answer page after page of questions. When students reach the last page of the quiz—maybe a page summarizing their results—you could automatically log them out. The next student sitting down at the same computer would have to log in, preventing the testing application from thinking the new test taker is the same person as the previous student.

   The second method lets visitors log themselves out by clicking a link, so the menu starts out reading, "Create new link: 'Log out'," which adds a new link with the words "Log out" to the page. After adding the behavior, you can then edit the page and change *Log out* to any text you like, or even add a graphic button to the link.

Tip: You can also first add some text like "Quit system," select it, and then apply the Log Out User server behavior. Dreamweaver will automatically use that text, instead of its standard "Log Out" text, when creating the link.

5. **Click Browse; navigate to and select a page from your site.**

   Good choices for this page are the login page—so the next visitor can log in—or the home page.

6. **Click OK.**

   You've just applied the link and server behavior.

Protecting Individual Pages

To password-protect a Web page, apply the "Restrict Access to Page" server behavior. You have to do this for each page you wish to protect, and you can only apply it to dynamic Web pages. In other words, you can't password-protect regular HTML files, text files, graphics, or any other file that isn't first processed by the application server.

Note: Although some Web servers let you password-protect an entire folder's worth of files, Dreamweaver doesn't provide any tools to do so. (If your site runs on an Apache Web server, however, you can use .htaccess files to password-protect an entire folder. You'll find a quick tutorial at *www.freewebmasterhelp.com/tutorials/htaccess/3*, and a free online tool for creating these files at *www.webmaster-toolkit.com/htaccess-generator.shtml*. Visit *http://apache.org/docs/howto/htaccess.html* for more information.)

The "Restrict Access to Page" behavior works like this: When someone attempts to load a password-protected page, programming code in the page determines whether he's already logged in. If the page also requires a particular access level—administrators only, for instance—it checks to see whether the visitor has the proper clearance as well; if so, the browser displays the page. If the visitor isn't logged in, however, or doesn't have the proper access level, then he's redirected to another page—like an "Access Denied" page or back to the login page.

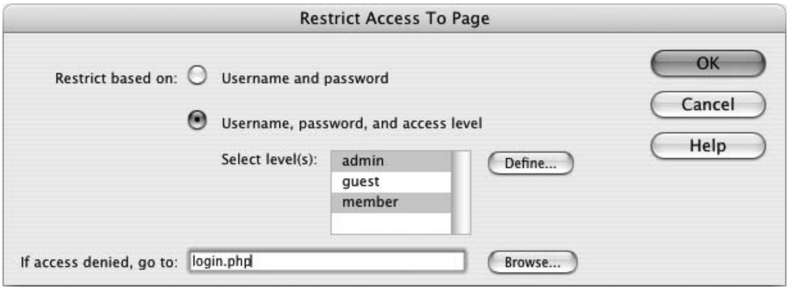
To apply this server behavior, follow these steps:

1. **Open the dynamic page you wish to protect.**

   It must be a dynamic page that uses the site's server model (see page 751).

2. **Open the Server Behaviors panel (Window → Server Behaviors). Click the + button and choose User Authentication → "Restrict Access to Page".**

   The "Restrict Access to Page" window appears (see Figure 25-5).



Restrict Access To Page

Restrict based on: ○ Username and password

● Username, password, and access level

Select level(s): admin, guest, member

If access denied, go to: login.php

Figure 25-5:
If you want to give access to more than one group, you can Ctrl-click (⌘-click) more than one level in the Select Levels list to highlight them simultaneously.

3. **Turn on one of the two radio buttons.**

   If you want to allow access to anyone in the users table, then select the Username and Password button. However, if you want to limit the page to visitors with a particular access level, then turn on the second button.

The first time you use this behavior, you'll have to define the different access levels, so click Define. You must type in each access level exactly as it appears in the database—*admin*, *member*, and *guest*, for example. Capitalization counts.

You need to define these access levels only once. Dreamweaver will remember these settings for other dynamic pages in the same site.

4. **Click Browse; navigate to and select the page people will see if they aren't logged in.**

   It's often a good idea to simply dump unregistered visitors onto the login page. That way, if they're legitimate customers, they can simply log in and return to the page. (Dreamweaver can help with this. See step 8 on page 892.)

5. **Click OK to apply the link and server behavior.**

Like the other server behaviors, Dreamweaver lists the "Restrict Access to Page" behavior in the Server Behaviors panel after it's applied. If you wish to change any of its properties, double-click its name in the panel. To delete the behavior, select it in the Server Behaviors panel and then click the minus (-) button.

Additional Data Sources

So far, you've been using Dreamweaver's dynamic page-building features to retrieve information from databases to build catalog pages, product detail pages, and other pages that require database-generated content. But at times, you'll want to collect data from other sources and add them to your page. For example, when someone logs into a site (see the Log In User server behavior on page 891), her user name travels along with her from page to page in what's called a *session variable*. Using the Bindings panel, you can capture this session name and then use it on a Web page.

Similarly, you can create *cookies* to store small pieces of information on a person's computer—such as a counter tracking how many times he's been to your site—and use Dreamweaver's Bindings panel to add that information to a Web page.

The Bindings panel lets you access these sources of data, as well as information submitted from form fields and embedded in URLs. Each server model understands different types of dynamic data, but most recognize the ones listed below.

Note for ASP.NET Users: Dreamweaver doesn't let you add any of these additional data sources to the Bindings panel.

Regardless of the type of dynamic data you wish to add, the process of accessing these data sources is essentially the same. It differs only among server models.

For PHP and ColdFusion

1. **Click the + button in the Bindings panel and then select the proper variable type: URL, Form, Cookie, or whatever (see Figure 25-6).**

   A window appears for the particular type of data source.

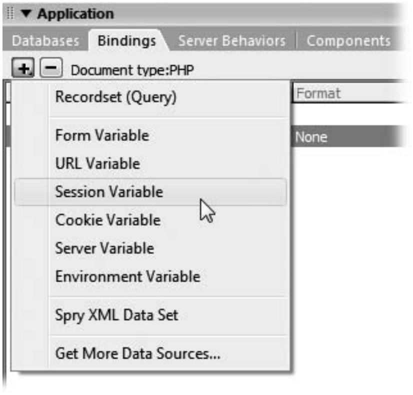


Figure 25-6:
Recordsets aren't the only type of data you can add to the Bindings panel. You can add the names of cookies, session variables, form fields, and other sources of data to the Bindings panel, then drag them onto the page. The XML Dataset option is new in Dreamweaver CS3 and is part of the Spry data tools discussed on page 476.

2. **Type the name of the variable in the Name field.**

   Capitalization matters; *username*, *UserName*, and *USERNAME* are all different variables. Find a system you're comfortable with (all lowercase, all uppercase, or mixed case) and stick with it.

3. **Click OK.**

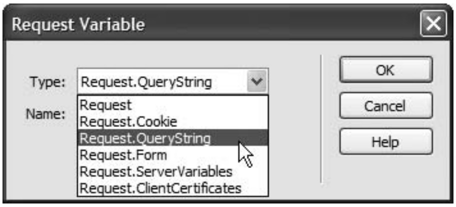
   Dreamweaver adds the variable to the Bindings panel.

Note: These steps don't actually *create* the variable; they only let you find out what's stored in a variable that's already been created, and then use it on a Web page. For example, adding a cookie variable in the Bindings panel doesn't actually create a cookie on a visitor's system. (For information on creating cookies, see page 901.)

For ASP

1. **Click the + button in the Bindings panel. Depending on the type of dynamic data you're interested in, select either Request Variable, Session Variable, or Application Variable.**

   In ASP, Request Variable covers a wide range of data sources, including form variables, URL variables, cookies, and server variables, so there's an extra step you must perform. After selecting Request Variable to open the Request Variable window (see Figure 25-7), choose a type of variable from the Type menu.


Request Variable

Type: Request.QueryString

Name:

Figure 25-7:
The Request Variable window (for the ASP server model only) lets you add a wide variety of variable types for use in an ASP page. "Request.QueryString" is ASP's way of referring to a URL variable.

2. **Type the name of the variable into the Name field.**

   Capitalization doesn't matter. To ASP, *username*, *UserName*, and *USERNAME* are all the same.

3. **Click OK.**

   Dreamweaver adds the variable to the Bindings panel.

For JSP

For JSP pages, Dreamweaver supplies tools for adding only form, URL, and session variables:

1. **Click the + button in the Bindings panel and then select either Request Variable or Session Variable.**

   Depending on the choice, either the Request Variable or Session Variable dialog box appears.

2. **Type the name of the variable into the Name field.**

   Here again, capitalization matters, so *username*, *UserName*, and *USERNAME* are all different variables.

3. **Click OK.**

   Dreamweaver adds the variable to the Bindings panel.

After you add the variable to the Bindings panel, you can drag it to your Web page; any of the techniques for adding recordset data to a page also work (see page 800), and you can use them whenever the Dynamic Data window appears (see Figure 24-7), for example, as the content of a form field.

Tip: You can drag data sources listed in the Bindings panel into Code view, as well. Once you've got your programming chops sharpened, this trick is helpful for quickly adding data to your own server-side programs.

URL Variables

Some URLs include information tagged onto the end of the name of a Web page, like this: *http://www.cosmofarmer.com/product.php?productID=10&action=delete*. The information following the ? is known as a *query string*, and it provides additional information to a dynamic page.

In most cases, this information comes in the form of one or more name/value pairs, which Dreamweaver refers to as *URL variables*. In this example, there are two URL variables: the first is named *productID* and its value is *10*; the second is named *action* and its value is *delete*. URL variables are often used to transfer specific information for use in a recordset. You saw an example of this in the tutorial in Chapter 23: the number of a particular product was passed in a URL to the product details page, which used this number to retrieve details on a particular item.

You can also add a URL variable to the Bindings panel, and then include it in a Web page or use it anywhere you'd use a dynamic data source. For example, you can use it as a parameter added onto the end of a link to hand off the information to another page.

Keep in mind that a page that links *to* the page using the URL variable must include the proper query string in the link. For example, say you add a URL variable named "username" to the page *crop_circles.html*; the page uses the query string to personalize the page: "Welcome, [username]". For this to work, you then need to link to the *crop_circles.html* page with the query string attached to the URL, like this: *crop_circles.html?username=bob*.

You can add a URL variable to a link using the methods described on page 790.

Tip: Don't use this method for accessing private or sensitive data. For example, suppose you used a URL variable as a method for accessing the personal data of one of your customers, like this: *customer_data.asp?customerID=78*. A nefarious visitor could just change the number in the URL to, say, 79 to view all of the personal data for customer number 79.

Form Variables

You can also add information from *forms* to the Bindings panel and use them on your page. If you add a form on one page, you can then collect that information on the page the form submits to (the same page specified in the form's *Action* property, as described on page 402). In other words, the page receiving the form data can display that information on the page *or* use it in some other fashion—such as inserting the information into the database, or creating a cookie or session variable.

If you're mainly using forms in conjunction with Dreamweaver's Insert Record and Update Record server behaviors, you won't generally take advantage of form variables. Those two behaviors work by collecting data from a form, adding or updating a database record, and then redirecting the Web browser to another page. The page the visitor finally sees never has access to the form information, so you can't add any form variables to that page.

However, adding a form variable to the Bindings panel can come in handy when you create a search page. For example, suppose you've created a page for searching a database. The search form lets the visitor type in a name—of an author or musician, for instance. You could then create a search *results* page that looks in the database for any records that match the search term. On that page, along with the database results, you could add text like "Search Results for: [search_term]", where *search_term* would be the word the visitor typed into the form. Just add the form variable to the Bindings panel and then drag it to the spot in the search page where you wish it to appear.

Note: If you use the GET method for submitting a form, the names and values of each form field are included in the URL. In this case, they're considered URL variables, so if you wish to add any of these fields to the Bindings panel, use the URL variable method instead. (For the difference between GET and POST, see page 403.)

Cookies

One problem with Web servers is that they have no memory.

Suppose, for example, that a site has a particularly long and annoying Flash movie that welcomes visitors with an ear-pounding, seizure-inducing multimedia display. Even if the designer was kind enough to include a "Skip this nauseating display" button, the Web server won't remember that you clicked it the *last* time you were there.

To overcome this limitation, most Web browsers can store *cookies*—small text files containing specific information—that Web servers create and read. In the example above, the Web server could drop a cookie onto your computer when you click the "Skip intro" button. The next time you visit the site, the Web server reads the cookie and kindly ushers you past the Flash movie and directly to the home page.

You can use cookies to store information on visitors' computers, too. They're a great way to store customer ID numbers, the number of visits to a particular page, and other bits of identifying information.

Cookies play by a few rules:

- A single cookie is stored on just one browser and one computer at a time. If you log onto a site that adds a cookie to your computer, and then log on again later from the public library, that computer won't have access to the cookie. In fact, if you use a different Web browser on the *same computer*, the Web server won't be able to read the original cookie from the other browser. (A variation: In some corporations, a Web browser stores cookies on a network server. This kind of cookie *can* be accessed by a particular browser—Internet Explorer, for example—on different computers on the network.)
- Only the domain that created the cookie can read it. *You* can't create dynamic pages that read a cookie set by Amazon.com, for example. Fortunately, that means other Web sites can't read the cookies you set on your visitors' computers, either.
- Web browsers limit the size of a cookie to 4 KB and only allow a limited number of total cookies (usually 300), per computer so that hard drives won't crumble under their weight.

You can add a cookie to the Bindings panel using the method described on page 897. Unfortunately, Dreamweaver doesn't provide any tools for creating cookies (you can submit feature requests for the next version of Dreamweaver on the Adobe Web site at *www.adobe.com/cfusion/mmform/index.cfm?name=wishform*). Several third-party developers have risen to the occasion, however:

- **PHP developers.** Dreamweaver Extension developer Felice Di Stefano has developed a free cookie extension for the PHP server model. It includes server behaviors for adding and deleting cookies from a PHP page. In addition, it can set a cookie to the value of a form field, or redirect a visitor to another page if a specific cookie doesn't exist, or if it matches a particular value. You can find it at the Adobe Exchange (page 738) or at *www.felixone.it*.

POWER USERS' CLINIC

Adding and Deleting Cookies Using PHP Pages

While Dreamweaver doesn't provide a tool for adding the scripts necessary to create and delete cookies with PHP pages, it isn't difficult to add the code yourself. (Dreamweaver can easily retrieve cookie information, as described on page 901.)

First, decide which page should add the cookie. The script will run when a visitor's browser *requests* the page, sending the cookie to the browser before the page content. Thus, you could add a script like this at the beginning of a page that receives and processes form information. For example, if someone registers at your site, your script can store the name he enters in the registration form as a cookie on his computer. When he returns to the site, the home page can then read the cookie and display a message like "Welcome back, Bob."

To add a cookie, put the following code above the <!DOCTYPE> declaration in the HTML code of the page.

```
<?php setcookie("name_of_cookie", "value_
of_cookie", time( )+2419200); ?>
```

Remember to include the opening <?php and closing ?>, which tell the application server that everything in between is programming code and not HTML. Replace *name_of_cookie* with whatever name you wish to give the cookie: *username*, for example. Also replace *value_of_cookie* with whatever you want to store in the cookie. In many cases, this will be a dynamic value–information from a recordset, a URL variable, or a form variable, for example. Using the steps described on page 897, add the appropriate dynamic data to the Bindings panel, and then drag it into the code, replacing the text (including the quote marks) "value of cookie".

Finally, you can set the amount of time the cookie stays on the visitor's computer. That's the *time( )+2419200* in the code above. Essentially you're saying that the cookie should stick around for a certain number of seconds (2419200) after the current moment (time( )). In this case, 2419200 is 30 days or about 1 month. If you wanted the cookie to stick around for an hour use 3600; for 1 day, use 86400.

PHP is a little persnickety about where you place this code: It has to come before all the code in the page, with the exception of other PHP code. If there's even just a single blank line (not within the <?php ?> tags), you'll end up with the much dreaded "Headers already sent" error.)

You can, however, place the code *after* other PHP code at the beginning of the file. For example, if you want to set the value of a cookie using information retrieved from a recordset, then you need to place the cookie code *after* the recordset code.

You may also want to delete a cookie at some point. For example, on an e-commerce site, you could use a cookie to store items a visitor adds to her shopping cart. When she wants to empty her cart–after she purchases everything in it, for example–you could simply delete the cookie. Just assign no value and a time in the past (no kidding) to the cookie you wish to delete, like this:

```
<?php setcookie('name_of_cookie', '',
time( )-3600); ?>
```

- **JavaScript cookies.** You can also use JavaScript to set cookies. This technique works with any type of page—even nondynamic pages. The only catch is that the visitor's browser must both understand JavaScript and have JavaScript enabled (most do). Dreamweaver comes with two Snippets for setting and reading cookies using JavaScript. They're in the cookies folder of the JavaScript folder in the Snippets panel. See Chapter 18 to learn about Snippets. For the king of JavaScript cookie creators, check out WebAssist's Cookies Toolkit at *www.webassist.com/professional/products/productdetails.asp?PID=109* (this is a commercial product that runs around $50 and also includes tools for adding cookies using server-side tools for PHP, ASP, and ColdFusion).

Session Variables

Web servers don't know or care whether the person requesting your company's home page just placed a $10 million order or is a first-time visitor. Of course, *you* probably care, and so do most Web applications, which need to follow visitors as they travel through a site. For example, in a typical e-commerce site, people use a "shopping cart" to store items they're interested in purchasing. For this to work, however, you need to track each shopper's movement from page to page.

To make this possible, most Web servers recognize what are called *session variables*. A session variable is created by the Web developer (or, more accurately, by a dynamic Web page that creates the variable) and follows the visitor from page to page. This type of variable lasts, logically enough, for a single *session*: if the visitor closes the browser, the session ends and the variable disappears. Most Web servers also add a limited time that the variable sticks around—usually 20 minutes. In other words, if the visitor doesn't hit any page in the site for 20 minutes, the Web server assumes that he's no longer around and destroys the session variable.

Note: Session variables take up resources from the Web server. That's why a Web server gets rid of them as soon as it can. Creating lots of session variables for a busy Web site can slow down the server.

When it creates a session variable, the Web server sends a cookie to the visitor's machine. The cookie contains a unique number (not the actual data contained in the variable), which the server uses to keep track of each visitor. When that person requests a page, the Web server reads the cookie with the unique ID. It can then retrieve session variables for that individual. For this reason, session variables won't work if the visitor's Web browser doesn't accept cookies. (PHP, however, has a built-in method for maintaining session information even when cookies aren't turned on.)

Note: Dreamweaver itself creates session variables when you use the User Authentication server behaviors. See page 894 for a discussion of these session variables and how you can use them.

You may be wondering how cookies and session variables differ, and when you'd want to use one over the other. The difference is that cookies can last between visits. If you want access to a piece of information when a visitor comes back tomorrow, or next week, or next month, use a cookie. For example, you'd use a cookie to remember a selection someone made from a previous visit, such as "Skip this crazy Flash Intro."

Session variables, on the other hand, provide better security. The actual information stored in the session variable stays on the Web server, while cookies exist as text files on a visitor's computer and can be opened and read by anyone with access to the computer. Accordingly, if you need to keep track of a confidential piece of information (someone's bank-account password, for example), you'd use a session variable.

POWER USERS' CLINIC

Adding and Deleting Session Variables Using PHP Pages

While Dreamweaver doesn't provide a simple wizard for adding the code necessary to create and delete session variables with PHP pages, it isn't difficult to add it yourself. (Dreamweaver does, however, make quick work of retrieving session variables; see page 903.)

The procedure is much like the one for adding cookies (see the box on page 902)—for example, here again, the script will run when a visitor requests the page. When people register at your site, therefore, the email address they enter in the registration form could be stored as a session variable.

To add a session variable, you must do two things. First, add this code near the top of the file:

```
<?php if (!isset($_SESSION))  session_
start(): ?>
```

This code simply alerts PHP that you wish to use session variables on this page (if you omit this line, you won't be able to set or read session variables). As when setting a cookie, there can't be any HTML or even empty space before this line or you'll get a "Headers already sent" error. Next, you set the session variable.

```
<?php $_SESSION['name_of_variable'] =
'value_of_variable'; ?>
```

Replace *name of variable* with whatever name you wish to give the session variable: *email*, for example. Also replace *value of variable* with whatever you want to store in the session variable. In many cases, this will be a dynamic value, like information from a recordset, a URL variable, or a form variable. Using the steps described on page 897, add the appropriate dynamic data to the Bindings panel, and then drag it into the spot in the code in the previous column just after the = sign (in this case, omit the set of quote marks: ''). As with cookies, *where* you place the session-creating code determines *when* it kicks in. So if you want to set a session with a value from a recordset, place the session code after the code that creates the recordset.

You may also want to delete a session variable to conserve server resources. (Dreamweaver's Log Out User server behavior uses this method to log a visitor out of a site.) To delete a server variable, add this code to the beginning of a page:

```
<?php unset($_SESSION['name_of_
variable']); ?>
```

To delete all session variables for a particular individual in one fell swoop, use this code:

```
<?php
if (!isset($_SESSION))
session_destroy();
?>
```

You can add a session variable to the Bindings panel using the method described on page 897.

Unfortunately, as with cookies, Dreamweaver doesn't provide any tools for creating or destroying session variables. To find third-party extensions that work with session variables, try the Adobe Exchange (*www.adobe.com/exchange*). Click the Dreamweaver link, and search using the term *session*.

Note for PHP Users: Felice Di Stefano has developed a free session extension for the PHP server model. It includes server behaviors for adding and deleting session variables from a PHP page. You can find it at the Adobe Exchange or at *www.felixone.it*.

Server Variables

Web servers collect and produce lots of information, much of which is hidden from the everyday Web surfer (and even the everyday Webmaster). Some of that information is obscure, but some can come in handy. For example, you can find out which Web browser the visitor is using, or which language the browser uses. While the exact list of server variables differs by Web server, here are some useful variables that work on many Web servers:

- **HTTP_USER_AGENT.** Information about the browser visiting the page. Unfortunately, you don't get a neat little summary like *Firefox 2 for Windows*. Instead, browser info is usually rather long-winded, like: *Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US; rv:1.8.1.2) Gecko/20070219 Firefox/2.0.0.2*. To decipher this confusing jumble of information, visit *http://en.wikipedia.org/wiki/User_agent#Example_user-agent_strings*.

- **REMOTE_ADDR.** The IP address of the computer requesting the Web page. It'll look something like 65.57.83.12. Depending on the visitor's setup, this could be the exact address of the computer. (Big Brother, where art thou?)

  Knowing this address has its uses. If someone frequently causes problems on your site—posts phony information to registration forms, say, or submits offensive messages to a message board—one potential solution is to prevent submissions to your database from that particular IP address. (However, since many users' IP addresses frequently change, this isn't a foolproof solution.)

- **HTTP_REFERER.** This is the URL of the page that *leads* to the current page. For example, say you click a link on page A to get to page B. Page B's HTTP_REFERER server variable would be A.

  You could use this knowledge to create the ultimate Back button. Simply add the HTTP_REFERER server variable to the Bindings panel. Then add a link to whatever you wish to be the Back button—graphic or text—and use the server variable as the link. When visitors click this link, it will take them back to whichever page brought them there in the first place.

For a list of server variables for use with the Apache Web server (the server most commonly used with PHP), visit *www.php.net/reserved.variables*. For a list of server variables supported by Microsoft's IIS Web server, visit the Microsoft Developer's Network site at *http://msdn2.microsoft.com/en-us/library/ms524602*.

Advanced Server Behaviors

In addition to the server behaviors described already, you'll find two other sets of behaviors that come in handy on dynamic Web pages.

Tip: You can download many more third-party server behaviors from the Adobe Exchange. Click the + button in the Server Behaviors panel and choose Get More Server Behaviors. Dreamweaver launches your Web browser and connects to the Dreamweaver Exchange site.

Not all extensions listed here work with Dreamweaver CS3. On the other hand, many of the server behaviors for Dreamweaver 8 also work with Dreamweaver CS3.

Recordset Paging

This set of five different behaviors lets you add links for jumping to different records in a recordset (straight to the last record, for example). In fact, Dreamweaver makes use of these same behaviors as part of its Recordset Navigation Bar object (page 807). You'll use these for moving through a long list of records, like a complete listing of products in a database.

To begin, add a recordset to a page. It should contain multiple records, since jumping to the *next* record when there's only one doesn't make much sense. The page could also contain a repeating region, so that several records appear.

You can add the recordset-paging server behaviors from the Server Behaviors panel, or from the recordset-paging menu on the Application tab of the Insert bar (see Figure 25-8):

- **Move to First Page** adds a link that jumps to the first record in the recordset.

Note: In ASP and JSP, you'll see the word "Record" instead of "Page" in this example and the following ones. For instance, you'll see "Move to First Record" instead of "Move to First Page".

- **Move to Previous Page** adds a link that jumps to the record before the current record. If you use a repeating region, it jumps to the previous *set* of records. For example, say you create a repeating region that displays five records at a time. If the page currently displays records 6–10, clicking a link with this server behavior applied to it causes records 1–5 to appear.

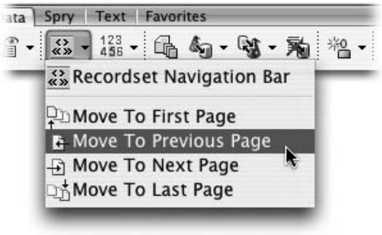


Figure 25-8:
You can create your own recordset navigation controls using the recordset-paging server behaviors.

- **Move to Next Page** adds a link that jumps to the next set of records in the recordset
- **Move to Last Page** adds a link that goes to the last set of records in the recordset.

- **Move to Specific Record** adds a link that goes to a single record based on information passed in the URL. (This option is available only in the ASP and JSP server models.)

Using any of the first four behaviors involves the same steps:

1. **Create a recordset and add dynamic content to the page.**

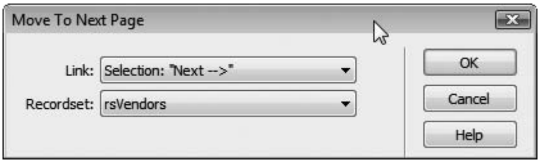
   For example, you could create a list of all the products your company sells. The recordset should contain at least enough records to span several pages. (You wouldn't use any of these behaviors if you displayed *all* of the records on a single page.)

2. **Click the page where you wish to insert the link.**

   You can also select an item on the page—text or a graphic—that you'd like to turn into a link.

3. **From the Server Behaviors panel, click the + button. Select Recordset Paging, and then choose a behavior from the submenu.**

   The window for the particular server behavior appears (see Figure 25-9).


Move To Next Page

Link: Selection: "Next -->"

Recordset: rsVendors

Figure 25-9: *The recordset-paging behaviors can add a new link, or add a link to text or an image you've selected on the page. You can also use the menu to select any link already on the page. That's usually not a good idea, however, since Dreamweaver will erase whatever link was previously applied.*

4. **Choose a recordset from the Recordset menu.**

   This is the recordset the behavior will move through.

5. **Click OK.**

   Dreamweaver adds the server behavior to the page and adds its name to the Server Behaviors panel.

The "Move to Specific Record" behavior, available in the ASP and JSP server models, works a bit differently. It doesn't add a link to the page; it simply forces the page to display a particular record when the page is requested. It does this by looking for a URL parameter that specifies a particular record in the recordset. In general, this isn't a good approach, since it requires the database server to retrieve many records just to display a single item from the database, which is a waste of server resources. On a busy site, that could mean a slowdown. You're better off creating a detail page (see page 814), which simply retrieves one record from the database.

Show Region Server Behaviors

At times, you'll want to display different information on a page based on the results of a recordset. For example, if a visitor searches your site's database of products for a product you don't sell, the search results page should say something like, "Sorry, we don't carry alligator skin bicycle seats." But if someone searches for a product you do sell, then the page should present the relevant details for that product. The Web page displays different text depending upon whether the search item was in the recordset.

Dreamweaver provides three sets of server behaviors that let you display any selection of HTML based on the results of a recordset (Figure 25-10):

- **Show If Recordset Empty.** If the recordset retrieves no records, this behavior will make the selected HTML appear in the browser window.

  This behavior comes in handy for a search page. On the search results page, apply it to some text like "We're sorry, your search retrieved no results," and you've got yourself a friendly solution for searches that turn up empty. This sever behavior is also handy for any page that displays detailed information about a single record, such as a detail page (page 814) or an update record page (page 849). Both of those depend on retrieving data from the database and are most often accessed using a URL variable like *product_details.php?productID=14*. If someone visits a page like that without the URL variable (for example, *product_details.php*) then they'll end up with a blank page. Use the Show If Recordset Empty behavior to list a message like "No product specified. Click here for a list of products."

- **Show If Recordset Not Empty.** If the recordset retrieves *any* records, this behavior will cause the selected HTML to appear in the browser window: a list of search results or details on a specific database record, for example. You'll often use this server behavior along with the Show If Recordset Empty server behavior. In this case, select the stuff you want to appear when a record is found: from the previous examples, you'd select all the HTML and code displaying the details of the record, or the record update form.

- **Show If First Page.** This server behavior, like the next three, works in conjunction with recordset-paging behaviors. It makes the selected HTML appear when the page displays the *first* record of a recordset, which comes in handy when you want to let people step through several pages of records.

- **Show If Not First Page** is the opposite of the previous one. If the page *does not* contain the first record in a recordset, then the selected HTML appears.

  Dreamweaver makes use of this behavior in its Recordset Navigation Bar (see page 807). In that case, if the page displays anything *except* the first set of records in a recordset, the First Item and Previous Page links appear. If the page *does* display the first item in a recordset, those links are hidden. (After all, you can't very well view the Previous Page if you're on page 1.)

- **Show If Last Page** functions just like the Show If First Page behavior, but for the last record in a recordset.
- **Show If Not Last Page** functions just like the Show If Not First Page behavior, but for the last record in a recordset. Dreamweaver uses this behavior to hide or show the Next Page and Last Item links in the Recordset navigation bar on the last page of records (page 807).

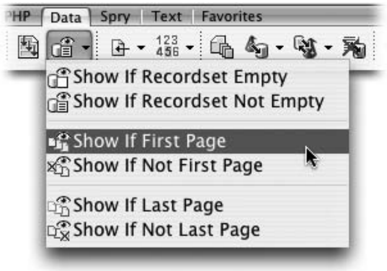


Figure 25-10:
The Show Region server behaviors are available on both the Application tab of the Insert bar and the Server Behaviors panel.

You can use these behaviors to show any selected object on the page—a paragraph of text, an image, a table, and so on. Your page can contain any combinations of these behaviors, and you can use any behavior two or more times to display multiple areas of a page. For example, maybe you'd like two things to appear when the recordset successfully retrieves a record: a graphic in the page's sidebar and a message in the main area of the page. You'd apply the Show If Recordset Is Not Empty server behavior twice; once for each selection of HTML (the graphic and the message).

You'll often use these behaviors in pairs. For example, a search results page should include both the Show If Recordset Empty behavior to display a "no results" message, and a Show If Recordset Not Empty behavior to display the search results.

To apply any of these behaviors:

1. **Create a dynamic page containing a recordset.**

   This could be a search results page or a master page that lists many records from the database.

2. **Select the HTML you wish to show based on a recordset outcome.**

   For example, when applying a Show If Recordset Is Empty server behavior to a search result page, select the message that should appear if the search returns no results.

3. **Open the Server Behaviors panel (Window → Server Behaviors). Click the + button, select Show Region, and choose one of the six behaviors from the submenu.**

   A window looking like Figure 25-11 appears. While the title of the window varies depending on which behavior you selected, each of the six behaviors has just this one option.

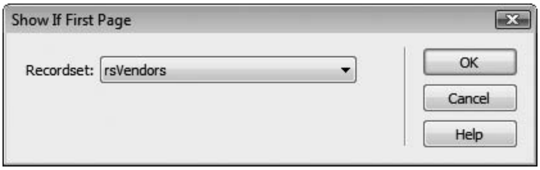


Figure 25-11:
Regardless of which of the Show Region server behaviors you apply, there's only one option to choose: the recordset whose results control the display of the region.

Note: The last four behaviors–Show If First Page, Show If Not First Page, Show If Last Page, and Show If Not Last Page–work only on pages that also have one of the recordset-paging server behaviors (discussed on page 906) applied.

4. **Select the name of the recordset from the menu and then click OK.**

   The recordset you select should be the one whose results you're interested in. For example, on a search results page, you'd select the recordset you created to perform the search.

After applying one of these behaviors to a selection of HTML, a gray line appears around the selection, and a gray tab appears bearing the words "Show If." That's the area that will appear if the given recordset condition is met (for example, if the page is displaying the last record of the recordset).

To remove a Show Region server behavior, select its name in the Server Behaviors panel and click the minus (-) button (or press the Delete key). Doing so removes the gray tab and outline. Now the affected HTML will appear regardless of the recordset results.

Note for PHP Users: An extension called PHP Show If Recordset Field Condition Is True (available for free on the Adobe Exchange site) lets you display part of a page when a field from a recordset matches a certain condition. Suppose, for example, that you have a products database with a field that records whether a particular item is for sale. If the item is indeed for sale, then you can use this behavior to display a large "For Sale" graphic on the product's detail page.

Tutorial: Authentication

In the tutorial at the end of Chapter 24, you created Web pages that could add, delete, and update records in the *CosmoFarmer* database. But you wouldn't want to allow just anyone who visits the Web site to access these pages, let alone delete products from the site. So in this tutorial, you're going to learn how to password-protect these sensitive, mission-critical Web pages.

The following steps assume that you've worked through the tutorial in Chapter 24 and have all of the completed files ready to go. You'll build on them in the following steps.

Building a Login Page

The first step is to create a login page—a simple form with fields for a user name and password. After a successful login from this page, an administrator will be able to access the administration pages:

1. **Open the file *login.php* in the admin folder of the site.**

   This page will contain the form for typing in an administrator's user name and password. You'll add the form next.

2. **Click in the empty space directly below the headline "Administrator Login." Choose Insert → Form → Form.**

   Dreamweaver adds a red dashed line to the page, indicating the beginning and ending <form> tags.

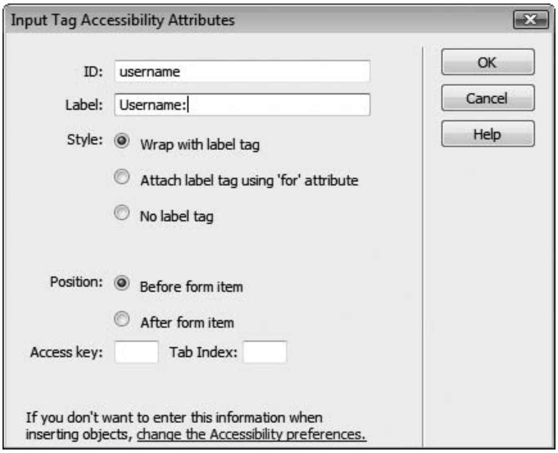
3. **In the Property inspector, type *login* as the name of the form.**

   While this step isn't required, it's good to get into the habit of assigning your forms descriptive names. Next, you'll add a box for entering a user name.

Note: The next steps assume you have the Form Accessibility feature turned on (see page 404). To make sure this is in fact the case, open the Preferences window by choosing Edit → Preferences (Dreamweaver → Preferences), select the Accessibility category, and make sure the "Form Objects" checkbox is turned on.

4. **Choose Insert → Form → Text Field.**

   The Input Tag Accessibility Attributes window appears (Figure 25-12). You'll add a label that will appear next to the form field on the page.



Input Tag Accessibility Attributes

ID: username

Label: Username:

Style: ● Wrap with label tag

○ Attach label tag using 'for' attribute

○ No label tag

Position: ● Before form item

○ After form item

Access key: Tab Index:

If you don't want to enter this information when inserting objects, change the Accessibility preferences.

Figure 25-12:
Dreamweaver's Accessibility features let you add helpful controls–including a descriptive label–to form elements. Setting the ID adds an ID property to the field (see step 3 on page 405), which is useful for adding CSS or using JavaScript to control the field. In addition, Dreamweaver uses the ID you supply for the field's name.

Note: You can also use the Forms tab of the Insert bar to add forms and form objects to your page, as described on page 401.

5. **In the ID box type *username*; in the Label box, type *Username:*; select the "Wrap with label tag" and "Before form item" buttons, and then click OK.**

   Dreamweaver inserts a text field with a descriptive label.

6. **In the Property inspector, choose Paragraph from the Formatting menu.**

   This wraps a paragraph tag around the label and form field. You now need to add a password field.

7. **Click to the right of the form field you just inserted and press the Enter (Return) key to add a new paragraph.**

   The routine for adding the next form field is the same.

8. **Choose Insert → Form → Text Field. In the window that appears, type *password* in the ID box, type *Password:* in the label box, and then click OK.**

   This inserts another form field and label.

9. **Select the new form field and, in the Property inspector, turn on the Password radio button.**

   By turning this form element into a password field, anything your visitors type in the field will be displayed like this **** or this ••••, hiding the secret password from nosy passersby watching over their shoulders.

   To complete the form, you'll add a Submit button.

10. **In the document window, click to the right of the password field and press the Enter (Return) key. Choose Insert → Form → Button.**

    The Accessibility window appears yet again. In this case, however, you don't need to add a label, since text will appear directly on the button.

11. **Click Cancel to close the Accessibility window and insert a Submit button. Select the new button and change its value to Login in the Property inspector.**

    The form is complete. Now it's time to let Dreamweaver do its magic.

12. **Choose Window → Server Behaviors to open the Server Behaviors window.**

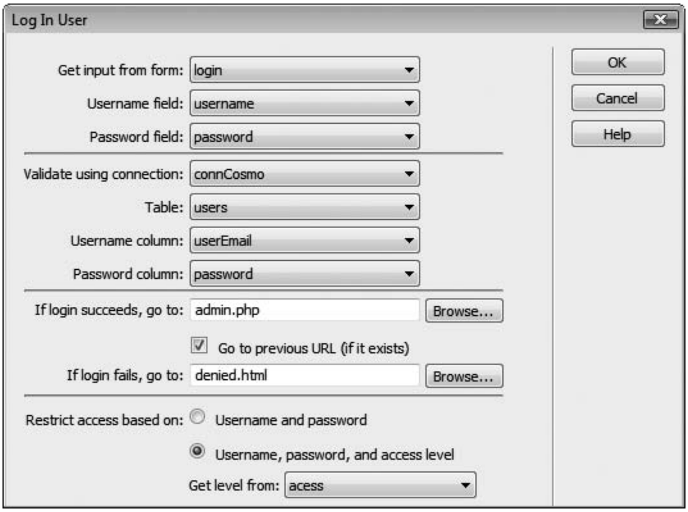
    Alternatively, you can use the keyboard shortcut Ctrl+F9 (⌘-F9).

13. **Click the + button and choose User Authentication → Log In User.**

    The Log In User window appears (see Figure 25-13). The first three items should already be filled out: the name of the form, the name of the user name field, and the name of the password field. If you had more than one form on the page, or additional fields inside the one form, you'd have to tell Dreamweaver which form and which fields to use for collecting the login information.

14. **From the "Validate using connection" pop-up menu, select "connCosmo".**

    This indicates which database contains the login information. You also need to specify which table and columns contain the user name and password.



Log In User

Get input from form: login

Username field: username

Password field: password

Validate using connection: connCosmo

Table: users

Username column: userEmail

Password column: password

If login succeeds, go to: admin.php

☑ Go to previous URL (if it exists)

If login fails, go to: denied.html

Restrict access based on: ○ Username and password

● Username, password, and access level

Get level from: acess

Figure 25-13:
Before visitors can access password-protected pages, they first need to log into the site. Dreamweaver's Log In User server behavior makes adding this feature a snap.

15. **From the Table menu, select Users. From the "Username column" pop-up menu, choose "userEmail". From the "Password column" pop-up menu, choose "password."**

    You've just established the basic logic of the login behavior: Whatever a visitor types into the two form fields is compared with data stored insides the Users table in the *CosmoFarmer* database.

    Next, you need to specify what happens when there's a match—when your visitor types a valid user name and password into the form—and what happens when the visitor types in an invalid user name or password.

16. **Click the Browse button to the right of the "If login succeeds, go to" field. In the Select File window, navigate to and select the file *admin.php* inside the admin folder. Click OK.**

    You've just chosen the page that will appear if the login is successful; your visitor's browser will display an administration page. (For the purposes of this tutorial, it doesn't matter if the "Go to previous URL" checkbox [see step 8 on page 892] is turned on.)

17. **Click the Browse button to the right of the "If login fails, go to" field. In the Select File window, navigate to and select the file *denied.html* inside the admin folder. Click OK.**

    This, of course, is the "access denied" page that will appear when somebody types in an invalid user name or password.

    Because this section of the site is for administrators only, you'll add an additional layer of security by restricting administrative pages using an access level as well as a password and user name. In this way, you'll also be able to have other password-protected pages—such as a special "paid subscribers" section—for registered visitors, without letting them access administrative areas of the site.

18. **Select the "Username, password, and access level" button. From the "Get level from" pop-up menu, choose "access".**

    The database table includes a special field for defining the access privileges of each registered member. For example, each administrator record in the *users* table also includes the value *admin* in the Access field.

19. **Click OK. Save this file. Press F12 (Option-F12) to preview it in your Web browser.**

    You'll now try out your newly created login page.

20. **In your Web browser, type anything you want in the two fields; click Submit.**

    Unless you've just made an incredible guess, you just typed in a user name and password that doesn't exist in the database. If the technology gods are smiling, an "Access Denied" page appears.

    Now try it again.

21. **Click the "Try logging in again" link to return to the login page. Type *dibble@cosmofarmer.com* in the Username field and *sesame* in the Password field; submit the form.**

    This time, you're in; the browser takes you to the main administration page. Here, you can jump to the pages you created earlier for adding, updating, and deleting products.

Tip (Important!): *sesame* is simply an awful password. Don't ever use it, or any word you can find in a dictionary, as a password. The reason? Web vandals often launch so-called "Dictionary attacks," in which they run through different terms pulled from a dictionary until they find a match.

The login script works just fine—you end up at the right page when you type in a valid user name and password. However, none of those other pages are protected yet. You can go to any of them, even if you haven't logged in. In the next part of this tutorial, you'll lock down each admin page, so only logged-in administrators can access them.

Password-Protecting the Administration Pages

The password-protection features offered by Dreamweaver require you to add a server behavior to each page you wish to protect:

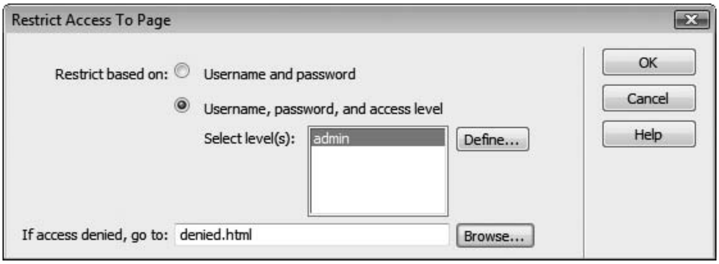
1. **Open the file *admin.php* in the admin folder.**

   This page is the main jumping-off point for adding, deleting, and updating products. It should be accessible only to administrators, so you'll add password protection to it.

2. **Make sure the Server Behaviors window is open (Window → Server Behaviors). Click the + button and choose User Authentication → Restrict Access To Page.**

   (Alternatively, you can use the User Authentication menu in the Insert bar, as pictured in Figure 25-1.)

The Restrict Access To Page window appears (see Figure 25-14). Since you want to limit access to administrators only, make sure the page is restricted to those with the proper access level.



Restrict Access To Page

Restrict based on: ○ Username and password

● Username, password, and access level

Select level(s): admin

If access denied, go to: denied.html

Figure 25-14: *Administrative pages can be reserved for those with the access level "admin," while regular subscribers to CosmoFarmer would have access to pages intended for subscribers.*

3. **Select the "Username, password, and access level" radio button.**

   You want to specify which type of user has access to this page, but first you must tell Dreamweaver what the different levels *are*.

4. **Click Define to open the Define Access Levels window. In the Name field, type *admin*. Next, click OK to close the window.**

   The word *admin* appears in the "Select levels" box. If you had other areas of the site with different access privileges, such as a subscriber area of the site only paying subscribers could access, you could continue to add levels by repeating this step.

5. **Click Browse; select the *denied.html* file in the admin folder and then click OK. Click OK again to close the Restrict Access To Page window.**

   To finish this page, you'll add a "Log out" link.

6. **Select Log Out (the last link in the left navigation bar on the page).**

   You'll turn this text into a "Log out" link.

7. **On the Server Behaviors panel, click the + button and then select User Authentication → Log Out User.**

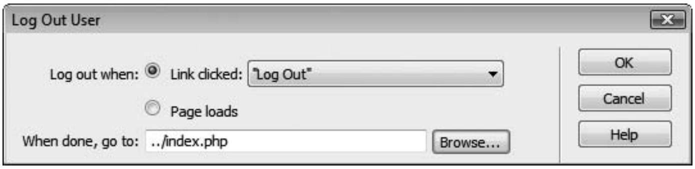
   Again, this option is also available from the Insert bar, as pictured in Figure 25-1. In any case, the Log Out User window appears (see Figure 25-15). The first radio button should already be selected. The text "Log Out" appears in the menu.

   These are the proper settings; you're simply adding the logout script to the words you selected on the page.

   Next, you'll tell Dreamweaver which page to go to after the visitor logs out.

8. **Click the Browse button; navigate to and select the file *index.php* in the root folder of the site, and click OK.**

When people log out, they'll simply end up at the main products page. Since they're no longer logged in as administrators, they won't be able to access any of the administrative pages without logging back in. The Log Out User window should now look like Figure 25-15.



Log Out User

Log out when: ● Link clicked: "Log Out"

○ Page loads

When done, go to: ../index.php

Figure 25-15:
This server behavior lets you offer visitors the polite option of logging out from your site. It destroys the session variables that track the login status of a visitor.

9. **Click OK to close the Log Out User window.**

   Now it's time to test the result.

10. **Choose File → Save; press F12 (Option-F12) to preview the page in your browser.**

    One of two things will happen: either you'll end up on the Access Denied page, or you'll see the *CosmoFarmer* administration page.

    If you quit your Web browser after the previous section of this tutorial, or never logged in to begin with, the Restrict Access To Page server behavior is working: It doesn't recognize you as a registered administrator and denies you access to this page. Click the "Try logging in again" link to go to the login page. Type *dibble@cosmofarmer.com* in the Username field and *sesame* in the Password field, and submit the form. You're now logged in and are taken to the admin page.

    However, if you logged in following the instructions from the previous section in this tutorial, and you haven't quit your Web browser in the meantime, you're still logged in. In this case, the Restrict Access To Page server behavior is again doing its job. You're allowed onto this admin page, because you *are* a registered administrator.

11. **Click Log Out.**

    The site logs you out and takes you to the main products page. To make sure you really are logged out, you'll open the administration page again.

12. **Return to Dreamweaver and the *admin.php* page. Press F12 (Option-F12) to preview the page again.**

    You're immediately redirected to the Access Denied page. You're not logged in, so you can't see the administration page. Hooray! The page is successfully protected from snoops.

Note: Some browsers will "cache," or store, the previously viewed administration page, so you might not actually see the Access Denied page. In this case, reload the page by clicking your browser's refresh button.

Of course, the most vulnerable pages (the update-, delete-, and add-product pages) are still accessible to everyone. You need to lock down those pages as well.

13. **Open the *add.php* page and repeat steps 2–5 on page 914.**

    Repeat this step for all other dynamic pages (*delete.php* and *edit.php*) in the *admin* folder, with the exception of *login.php*. (After all, *that page* should be visible to those who haven't yet logged in.)

    If you want, you can also add a Log Out link to each of these pages by repeating steps 6–9 on page 915.

Now all of the administrative pages in the site are password protected. Only authorized administrators who log into the site can add, edit, or delete records from the database.

Displaying a Portion of a Page to Logged-In Users

Even though unauthorized users can't access any of the pages that can change the database, they can still see the links you added to the Product Details page in the last chapter—"Edit this Information" and "Delete this Product." Nothing particularly earth-shattering will happen if they click them—unauthorized users will just end up at the Access Denied page—but even that's not very elegant. Wouldn't it be tidier if those links didn't even *show up* except to people logged in as administrators?

You'll set that up next:

1. **Open the *product.php* page.**

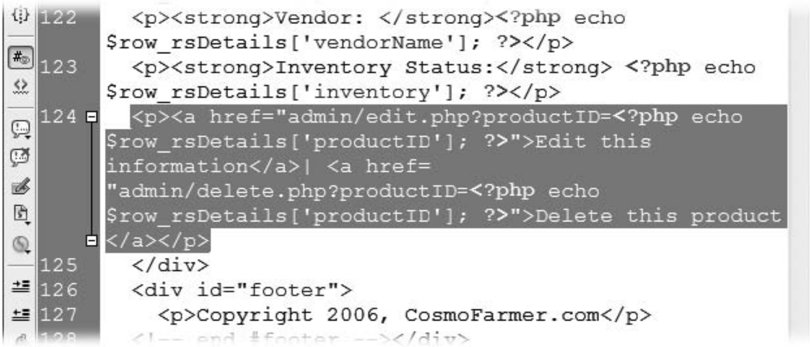
   You'll be doing a little painless programming in the Code view at this point.

2. **In the document window, click inside the text "Edit this Information"; in the Tag selector, click the <p>.**

   You've just selected the paragraph containing the two links. This paragraph should appear only to administrators.

3. **Choose View → Code.**

   The document window switches into Code view (see Figure 25-16).



```
  <p><strong>Vendor: </strong><?php echo
$row_rsDetails['vendorName']; ?></p>
  <p><strong>Inventory Status:</strong> <?php echo
$row_rsDetails['inventory']; ?></p>
  <p><a href="admin/edit.php?productID=<?php echo
$row_rsDetails['productID']; ?>">Edit this
information</a>| <a href=
"admin/delete.php?productID=<?php echo
$row_rsDetails['productID']; ?>">Delete this product
</a></p>
  </div>
  <div id="footer">
    <p>Copyright 2006, CosmoFarmer.com</p>
  <!-- end #footer --></div>
```

Figure 25-16: *When you enter Code view, whatever you had selected in the document window is highlighted in the code. The HTML code outlined here includes the Edit and Delete product links you wish to hide from unregistered visitors.*

4. **Click at the beginning of the selection, just before the opening <p>.**

   The insertion point is now at the start of the paragraph. You'll add some programming code here.

5. **Type *<?php if (isset($_SESSION['MM_UserGroup']) && $_SESSION['MM_UserGroup']=='admin') { ?>*.**

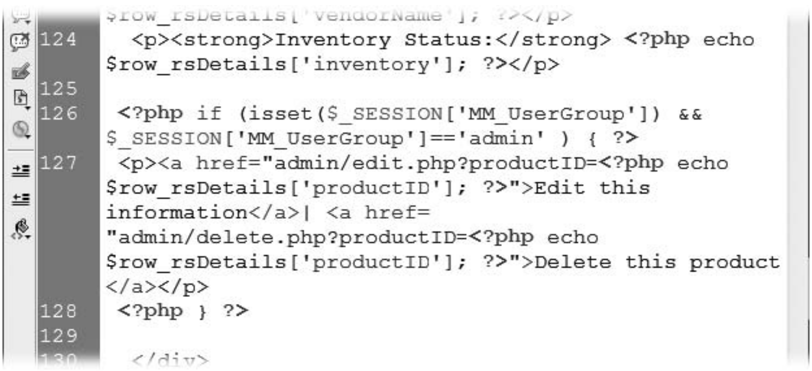
   The opening <?php tells the application server that some PHP code is coming. In other words, this isn't HTML—it's a program that the application server must process. The IF part of this code indicates what's called a *conditional statement*. In this case, it means, "*if* this person is logged in with an access level of 'admin', *then* the paragraph will appear on the page."

   To determine if the visitor is logged in with the proper access level, the code sneaks a peak at a session variable called MM_UserGroup. As mentioned on page 894, when someone logs into the site, the server behavior creates a session variable called MM_UserGroup. This variable follows the visitor around the site and contains a word that indicates what level of access he has. The programming you just added first checks to see if the session variable exists (that's the *isset* weirdness) and then verifies that the session variable is "admin", which indicates that the visitor is logged in as an administrator. If all of that is true, the paragraph letting the visitor access the edit and delete links appears.

6. **Click to the right of the closing </p> tag (just after "Delete this product") and type *<?php } ?>*.**

   This code concludes the conditional statement. In other words, all of the HTML between the first line of code you added in the previous step and this final *<?php } ?>* will appear *only* if the user is logged in as an administrator.

   The code should now look like Figure 25-17. You need to do one last thing on this page.



```
$row_rsDetails['vendorName']; ?></p>
  <p><strong>Inventory Status:</strong> <?php echo
$row_rsDetails['inventory']; ?></p>

 <?php if (isset($_SESSION['MM_UserGroup']) &&
$_SESSION['MM_UserGroup']=='admin' ) { ?>
 <p><a href="admin/edit.php?productID=<?php echo
$row_rsDetails['productID']; ?>">Edit this
information</a>| <a href=
"admin/delete.php?productID=<?php echo
$row_rsDetails['productID']; ?>">Delete this product
</a></p>
 <?php } ?>

  </div>
```

Figure 25-17: *Here's the finished code that hides the "Delete this Product" paragraph from unregistered users.*

7. **Stay in Code view, and scroll to the top of the page. Place your cursor after the very beginning of the file, before the very first line, which begins with <?php.**

   Next, you'll add a little code that lets the page access the session variables.

8. **Type *<?php session_start( ); ?>* and press return to move the existing code after this onto the next line. The first two lines in Code view should now look like this:**

   ```
   <?php session_start(); ?>
   <?php require_once('Connections/connCosmo.php'); ?>
   ```

   The code you just added makes PHP turn on its magical session-handling powers. Now this page will be able to "see" any session variables set for the current visitor—such as whether the visitor is logged in and if he or she is an "admin" user.

9. **Choose File → Save; press F12 (Option-F12) to preview the page.**

   Because you logged out earlier, the links should now be invisible. To see them, you must log in and return to the product details page.

10. **Go back to Dreamweaver and open the *login.php* page. Press F12 (Option-F12) to preview the page. Type *dibble@cosmofarmer.com* in the Username field and *sesame* in the Password field. Click Submit.**

    You're now logged in and taken to the main administration page. If you return to a product details page, the links will miraculously return.

11. **Click the Store button in the top navigation bar on the page to go to the product listings page. Click the name of any product to see its details.**

    Voilà! The links are back. You can freely edit or delete any product in the database. If you return to the administration page and click the Log Out button, you won't see these links until you log back in.

You could also use this trick to add "Log out" links to every page on the site, but only make them visible if the visitor is logged in. With very little programming experience, you can use Dreamweaver's server behaviors together (and perhaps bring in server behaviors from extension developers) to build sophisticated database-driven Web sites.

Now go forth and electrify your sites!

CHAPTER
26

Server-Side XML and XSLT

XML is everywhere. You'll find it used in countless files on your computer; for everything from tracking information in your iTunes music library to providing the structure and options in Dreamweaver's menus. On the Web, XML is used to broadcast newsfeeds, and provide product, pricing, and availability information from Amazon.com and eBay using a technology known as Web Services.

So what exactly is XML? XML, or Extensible Markup Language, is a tag-based language, somewhat like HTML, used to organize data in a clear, easy to understand way so that different computers, operating systems, and programs can quickly and easily exchange data.

Dreamweaver CS3's Spry XML Data Set tools (covered in Chapter 12) let you work with XML on the "client-side." Translation: someone visiting your site downloads a Web page, some Spry JavaScript programming, and an XML file; then, thanks to some fancy JavaScript magic, the Web page can display and interact with the XML data in a variety of ways. For example, you can publish a table of data that the visitor can sort dynamically, simply by clicking a particular column's header (see page 493 for more about this trick).

Note: For a detailed introduction to XML, go to page 477.

The tradeoff with the client-side approach is that it forces a visitor's Web browser to download the entire XML file. If the file is large, that can take a fair amount of time, since the browser downloads the whole enchilada (even the stuff you never intend to display). In addition, a Spry Data Set can only use an XML file that's stored on the same Web server as your Web page. In other words, you can't access the RSS feed (an XML format for broadcasting news, blog posts, and other information) of CNN.com or your favorite blogger.

Fortunately, you can use Dreamweaver's server-side XML and XSLT tools to overcome these limitations. The program's XSLT server behavior produces regular HTML out of XML and XSLT style sheets. (Hang in there: more in a moment on what XSLT is all about.) And, fortunately, since Dreamweaver handles all of the complex programming required to make this happen, it's no more difficult for you than building a dynamic Web page.

Understanding the Technologies

Although XML is very much like HTML in many ways, it doesn't have any inherent formatting capabilities. Unlike with HTML, where an <h1> tag is at least displayed differently—bolder and bigger—than other text, a Web browser doesn't know how to display an XML tag. XSLT and XPath are two complementary (and very complex) languages that let you define how XML tags should look. Fortunately, even though these languages are hard to master, Dreamweaver takes care of the entire process. All you need to know is how to use Dreamweaver's Design view to create cool-looking Web pages.

XPath provides the means to identify particular elements or tags in an XML file (see Figure 12-12 on page 483). In other words, it's like a set of directions for specifying a particular piece of information in an XML file.

XSLT is the magic dust that transforms an XML document into an HTML document. In fact, it's used to create any number of different types of documents for Web browsers, palmtops, printers, and so on, out of a single XML file. XSLT stands for Extensible Style Language Transformations, which is just a really weird name for a programming language that converts XML tags—<event>Halloween Social</event>, for example—into something else, like the code a Web browser understands—<h1>Halloween Social</h1>. In a nutshell, that's what Dreamweaver's XML tools do: They use XSLT to transform XML into HTML.

Note: Because XSLT adds formatting to XML, much like Cascading Style Sheets add formatting to HTML, you'll often see an XSLT file referred to as an *XSLT style sheet*.

Think of it this way: XPath is used to identify the XML tags that XSLT transforms into HTML. XSLT does the actual conversion to HTML, but XPath tells XSLT which tags to convert. They work hand in hand to get the job done. And, fortunately, that's all you need to know. In fact, it's more than you need to know to use Dreamweaver to turn XML into great-looking Web pages.

Tip: Dreamweaver includes built-in reference material covering XML and XSLT. See page 394 for more on Dreamweaver's Reference panel.

Creating Dynamic Pages with XSLT and XML

Dreamweaver's XSLT server behavior processes all of those "X" files and produces nothing but clean HTML for your visitors. To take advantage of this feature, you'll need to set up an application server, as described in Chapter 22, so that you can run ASP, PHP, JSP, .NET, or ColdFusion pages with the necessary programming code.

Next, you need to either have an XML file in your site, or know the URL of an XML file out on the Web that you'd like to use—for example, *http://www.cosmofarmer.com/feed.xml*. One option is to create an entire page that's an XSLT file—it will contain all the HTML for the general look of the page, and the XML information you wish to display. But this is generally an inefficient technique, since the server has to process the entire file (plain HTML and all) and you can't take advantage of Dreamweaver templates to enforce the look of your site.

Note for PHP Users: For server-side XSLT to work, the version of PHP you're using must have XSLT support. PHP 5 has this capability built in, but PHP 4 requires extra work to get this going. Unfortunately, many Web hosting companies still use PHP 4, and many of those don't offer XSLT support. So before moving ahead with your XML-fueled dynamic-page-creation efforts, call or email your Web hosting company to see if their PHP installation supports XSLT.

A better method is to create what's called an *XSLT fragment*, which lets you add a "chunk" of formatted XML to just one part of a dynamic page. For example, say you want to list the top 10 headlines from CNN.com's RSS feed on your home page. Using Dreamweaver, you can transform the newsfeed from XML into HTML code. Of course, that won't be the only thing you want on your home page. Most of the page will consist of information related to your site. In this case, you only need to dedicate a fragment of the page—like a sidebar on the right edge of the page—to these headlines.

Note: RSS and Atom (a competing standard) are simply two different XML-based formats used to identify document information–like an author's name, a title, or a brief article description–and provide a link to a complete article on a Web site. These formats are commonly used on news sites and blogs to provide readers with a syndicated news feed. RSS stands for "Rich Site Summary" or "Really Simple Syndication" (depending on whom you ask). Atom is a newer standard that competes with RSS but pretty much does the same thing. For more information on RSS, see *www.w3schools.com/rss/default.asp*, and for Atom, see *www.atomenabled.org/developers/syndication/*.

The process for creating and inserting an XSLT fragment is simple: create the fragment, add and format XML information, open a dynamic page, and then insert the XSLT fragment into it. When your visitors view the dynamic page, the application server will process the XSLT fragment and add its contents to the rest of the page.

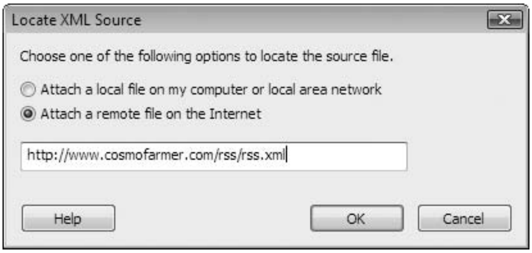
Here's how to create and use an XSLT fragment:

1. **Choose File → New.**

   The New Document window appears.

2. **Click the Blank Page button on the left side of the window. From the Blank Page list, select XSLT (Fragment), and then click the Create button.**

   The Locate XML Source window appears (Figure 26-1). Because XSLT is meant to make an XML file look great, you need to tell Dreamweaver which XML file to use. You have two choices when working with server-side XSLT: select a local file or type the URL of an XML file out on the Web.



Locate XML Source

Choose one of the following options to locate the source file.

○ Attach a local file on my computer or local area network

● Attach a remote file on the Internet

http://www.cosmofarmer.com/rss/rss.xml

Figure 26-1:
You can use either an XML file stored locally on your own site, or type the absolute URL of an XML file on the Web, such as the location of an RSS feed from a blog or news Web site.

3. **Select either "Attach a local file…" or "Attach a remote file on the Internet."**

   If the XML file is on your site, choose the first option. Select the second option if the XML file is on another Web site.

4. **If you're using an XML file on your site, click the Browse button to locate the file. Otherwise, type an absolute URL—*http://www.the_site.com/xml_file.xml*, for example—in the box. Click OK.**

   If you're pointing to a file out on the Internet, you *must* use a full, absolute URL including the *http://* part (Dreamweaver helps you out by adding *http://* to the box when you select the "Attach a remote file" button). Dreamweaver finds the file in your local site (or looks for the file out on the Internet), reads its contents, and displays the file's tags and properties in the Bindings panel. At this point, jump to page 926 to learn how to add XML data to the page.

   Although Dreamweaver claims that it's "attaching" the XML file to the XSLT document, it's really just adding a comment tag to the XSLT file, like this: <!--DWXML-Source="news.xml" -->. This comment helps Dreamweaver know which XML file to use with the XSLT document—so don't delete it. Technically, you actually attach an XSL file to an XML file to make this whole process work (as described below).

Note: If you want to try this out for fun, you can load an XML file from Wired.com. Use this URL: *http://www.wired.com/news/feeds/rss2/0,2610,,00.xml*.

5. **Save the new XSLT style sheet fragment.**

   Make sure the file ends in the extension .xsl. In addition, make sure you save it in the same folder as the dynamic page you wish to attach the XSLT style sheet fragment to. Otherwise, if the style sheet contains links, graphics, and other linked elements, they may not show up when the XML file is viewed.

Note: One way to get around this limitation of needing to store everything in the same folder is to use root-relative or absolute URLs (see page 153) for links, and to add graphics and external CSS files to your XSLT style sheet. Of course, taking these steps is probably more work than simply saving the XSL file in the same folder as your dynamic Web page.

6. **Add XML elements to the XSLT style sheet as described on page 926.**

   You can also add regular Web page content—like images, tables, CSS styles, and so on—to the page, and format the XML just as you would text on any other dynamically generated page. Once you've finished the design of your XSLT fragment, you then add it to your dynamic Web page.

Note: Because the XSLT fragment will be part of a larger Web page, you won't be able to see the effects of that page's CSS styles as you format your XML data. Fortunately, if you're using external CSS style sheets, you can use Dreamweaver's Design Time Stylesheets feature to temporarily attach, preview, and use the same CSS styles you're using on your final Web page. See page 307 for instructions.

7. **Open the dynamic page that you wish to add the XSLT fragment to.**

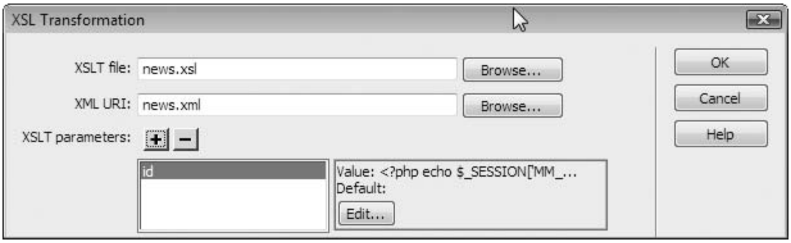
   This must be a dynamic page using the same server model as your site—for example PHP, ASP, or ColdFusion. Because the XML transformation magic occurs via programming that Dreamweaver inserts in the page, you can't add an XSLT fragment to a regular Web page (an .html file).

8. **Click where you wish to insert the XSLT fragment.**

   The spot you pick could be inside a table cell or within another layout region—such as a sidebar—on your page. The XSLT fragment will be added to this spot, just like a Dreamweaver Library item (see page 653)—that is, it will just be a chunk of HTML inside your page.

9. **Make sure the Server Behaviors panel is open (Window → Server Behaviors), click the + button, and then select XSL Transformation.**

   The XSL Transformation window appears (see Figure 26-2). Next, you select the XSL file you created earlier.



XSLT file: news.xsl

XML URI: news.xml

XSLT parameters: + −

id

Value: <?php echo $_SESSION['MM_...

Default:

Figure 26-2: *The XSLT Transformation window lets you attach an XSLT style sheet to a dynamic page. You can also send special information–XSLT parameters–to the style sheet that affects the display of the page (this process is described on page 939).*

10. **Click the top Browse button to open the Select XSLT File window. Navigate to and select the XSL file you created; click OK (Choose on the Mac) to choose the file, and then close the Select XSLT File window.**

    This tells Dreamweaver which fragment to use. In addition, Dreamweaver should automatically fill out the path to the XML file (it reads the comment inserted in the XSLT file identifying which XML file to use—see the Note below). If the XML file path doesn't appear, you can click the Browse button next to the XML file box and select the proper XML file yourself. In addition, if you're using XML from another Web site, you'll see the URL of that file, and the Browse button will disappear.

    The "XSLT parameters" option lets you pass information to the XSL file that can be used to alter how the XML file is displayed. This advanced feature is discussed on page 939.

11. **Click the OK button to close the window and insert the fragment.**

    Dreamweaver displays the XSLT fragment in your Web page. If you've set up a testing server (see page 754), you can preview the effect by pressing F12 (Option-F12).

You can't directly edit the fragment inside the dynamic Web page. Dreamweaver treats it like a single element on the page. To make changes to the fragment—to add graphics, change links, or reformat the XML—you must open the XSL file and make changes directly to it.

Note: Dreamweaver adds additional folders and files to your site when you use the XSLT server behavior. These additions contain the necessary programming code to successfully embed XML data into a Web page. That means when you're moving everything to your Web server–the dynamic page, the XSLT fragment file, and the XML file–you also need to upload these files, which you'll find stored in a folder named *includes* in your site's local root folder. Upload this folder to your Web server. (See Chapter 17 for instructions on using Dreamweaver's FTP tool.)

Inserting and Formatting XML

Now you know the basics of creating and using XSLT style sheets. But how do you actually add and format XML data? Dreamweaver makes this process easy, and if you've used the program's database tools, you already know how to do it: just use the Bindings panel. Once you create an XSL file and attach the XML file to it, Dreamweaver reads all of the tags in the XML file and adds them to the Bindings panel (see Figure 26-3).

You can drag any element in the Bindings panel into your XSLT style sheet page, just as you'd drag information from a database recordset. That means you can place XML information in a table cell, a footer, or a banner—anywhere you can place regular HTML elements on a page.

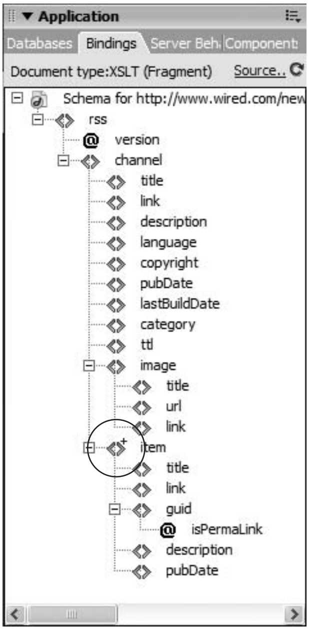
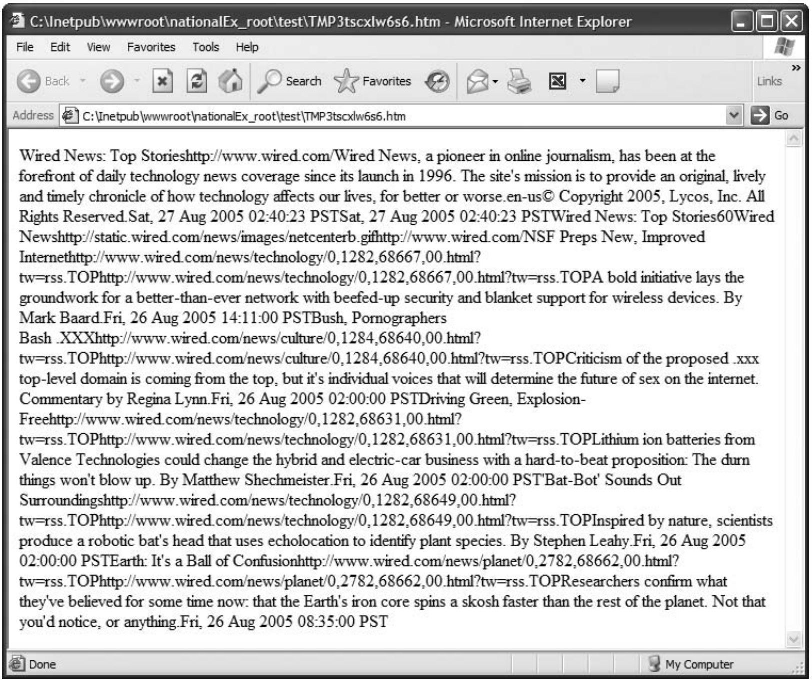


Figure 26-3:
When using Dreamweaver's XSLT tools, the Bindings panel lists all the tags and properties in the XML file you're formatting. Dreamweaver includes a few visual clues about the XML file: < > represents an XML tag and is the most common icon you'll encounter; the @ represents a tag property (also called an attribute; for example, in the tag <item id="154">, "id" is an attribute); and next to some tags, you'll see a small + (circled in this image) or a ?. The + indicates that the tag is repeated multiple times; in the XML for an RSS feed, each news item in the feed will have its own tag. The ? (not shown here) means the tag is optional, and it appears next to tags inside of other repeated tags (the ones with the +).

You should keep a couple things in mind when inserting XML using this dragging method:

- Only the contents of the XML tags and properties are inserted, not the tags or property names themselves. For example, in Figure 26-3, dragging the <title> tag that appears inside the <channel> tag onto a document just prints the text inside this tag, not the tag itself. This is a good thing: You don't usually want to include the tags: <title>An Important Story</title>. Instead, you just want the text: An Important Story.

- Dragging a tag that includes *other* tags often results in a hard-to-read mess. That's because Dreamweaver includes text from each of the nested tags, as well. For example, dragging the root element—"rss"—from the Bindings panel pictured in Figure 26-3 adds the simple label {rss} to the page in Dreamweaver's Design view. But when the page is actually viewed in a Web browser, you'll end up with one long paragraph composed of the text from all of the tags—the channel, the title, the description, and so on, as well as each of the repeated "item" tags as pictured in Figure 26-4. Dreamweaver treats this as a single big blob. To get around this, drag tags that don't include other nested tags. (Nested tags are called *child* tags.) For example, in Figure 26-3, the "title" tag that appears directly inside the "channel" tag doesn't have any tags inside it. Likewise, a repeating tag—item, for example—includes tags that don't have any children: "title," "link," "description," and "pubDate." These are all good candidates for adding to a document.


C:\Inetpub\wwwroot\nationalEx_root\test\TMP3tscxlw6s6.htm - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Links
Address C:\Inetpub\wwwroot\nationalEx_root\test\TMP3tscxlw6s6.htm
Go
Done
My Computer

```
Wired News: Top Storieshttp://www.wired.com/Wired News, a pioneer in online journalism, has been at the
forefront of daily technology news coverage since its launch in 1996. The site's mission is to provide an original, lively
and timely chronicle of how technology affects our lives, for better or worse.en-us© Copyright 2005, Lycos, Inc. All
Rights Reserved.Sat, 27 Aug 2005 02:40:23 PSTSat, 27 Aug 2005 02:40:23 PSTWired News: Top Stories60Wired
Newshttp://static.wired.com/news/images/netcenterb.gifhttp://www.wired.com/NSF Preps New, Improved
Internethttp://www.wired.com/news/technology/0,1282,68667,00.html?
tw=rss.TOPhttp://www.wired.com/news/technology/0,1282,68667,00.html?tw=rss.TOPA bold initiative lays the
groundwork for a better-than-ever network with beefed-up security and blanket support for wireless devices. By
Mark Baard.Fri, 26 Aug 2005 14:11:00 PSTBush, Pornographers
Bash .XXXhttp://www.wired.com/news/culture/0,1284,68640,00.html?
tw=rss.TOPhttp://www.wired.com/news/culture/0,1284,68640,00.html?tw=rss.TOPCriticism of the proposed .xxx
top-level domain is coming from the top, but it's individual voices that will determine the future of sex on the internet.
Commentary by Regina Lynn.Fri, 26 Aug 2005 02:00:00 PSTDriving Green, Explosion-
Freehttp://www.wired.com/news/technology/0,1282,68631,00.html?
tw=rss.TOPhttp://www.wired.com/news/technology/0,1282,68631,00.html?tw=rss.TOPLithium ion batteries from
Valence Technologies could change the hybrid and electric-car business with a hard-to-beat proposition: The durn
things won't blow up. By Matthew Shechmeister.Fri, 26 Aug 2005 02:00:00 PST'Bat-Bot' Sounds Out
Surroundingshttp://www.wired.com/news/technology/0,1282,68649,00.html?
tw=rss.TOPhttp://www.wired.com/news/technology/0,1282,68649,00.html?tw=rss.TOPInspired by nature, scientists
produce a robotic bat's head that uses echolocation to identify plant species. By Stephen Leahy.Fri, 26 Aug 2005
02:00:00 PSTEarth: It's a Ball of Confusionhttp://www.wired.com/news/planet/0,2782,68662,00.html?
tw=rss.TOPhttp://www.wired.com/news/planet/0,2782,68662,00.html?tw=rss.TOPResearchers confirm what
they've believed for some time now: that the Earth's iron core spins a skosh faster than the rest of the planet. Not that
you'd notice, or anything.Fri, 26 Aug 2005 08:35:00 PST
```

Figure 26-4:
If you use an XML tag that's too high up on the food chain–that is, one that has other tags nested inside of it–you can end up with a large chunk of hard-to-read text.

You can also insert XML into a Web page by choosing Insert → XSLT Objects → Dynamic Text or by clicking the Dynamic Text button on the Insert bar's XSLT panel (see Figure 26-5). Either method opens the XPath Expression Builder window (Figure 26-6). An XPath Expression is just a way of identifying a particular element—called a *node*—inside an XML file (see Figure 12-12 on page 483).

To add the dynamic text, select the XML tag or property you wish to insert. In the Expression box in the bottom half of the window, you'll see the XPath code required to locate your selection. For example, in Figure 26-6, the expression is *rss/channel/description*. This is shorthand for "Find the description tag, which is inside the channel tag, which is located inside the rss tag." In other words, this expression lists the order in which the tags are nested (in this sense, it's very much like the document window's Tag selector [see page 22]).

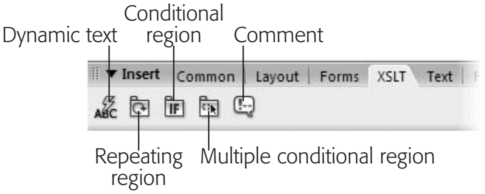
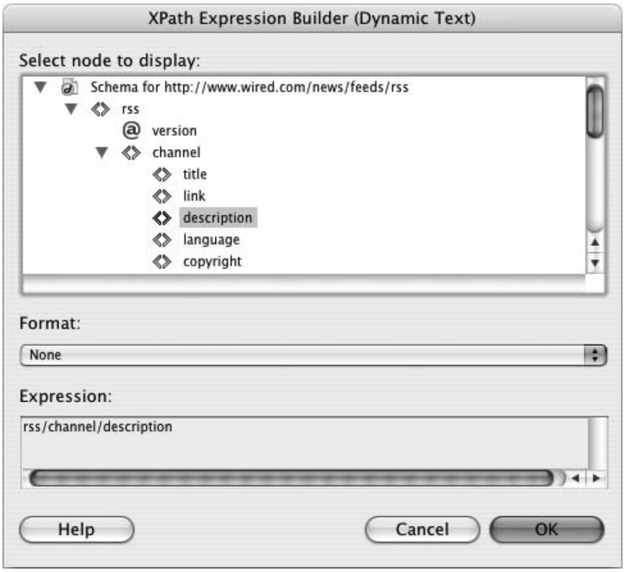


Figure 26-5:
Dreamweaver's XSLT tab includes five buttons for adding XSLT objects. The Comment object just inserts an XSL comment–like an HTML comment (see page 376)–so you probably won't use it much, if ever.



XPath Expression Builder (Dynamic Text)

Select node to display:

```
▼ Schema for http://www.wired.com/news/feeds/rss
    ▼ rss
        version
        ▼ channel
            title
            link
            description
            language
            copyright
```

Format: None

Expression: rss/channel/description

Figure 26-6:
Use the Format drop-down menu to apply a format to the XML data. Unfortunately, the formats are aimed almost entirely at formatting numbers–adding a $ sign to currency data, for example–so they won't do anything for text-only XML data.

Dreamweaver also lets you apply some formatting options to the selected text using the Format menu. Almost all of the options have to do with formatting numbers, so if you're inserting text that's actually a numeric value, these formatting controls can come in handy. For example, say you add a tag that's used to indicate a price: <price>3.25</price>. Selecting any of the currency options will add a $ sign in front of the number when it's displayed on the page. If you're dealing with big sums of money—<price id="Trump Tower">34589585</price>—then the "Currency group to 3 digits, 2 decimal places" is a good option. Then, the output would be something like this: $34,589,585.00. Again, all but two of the options are for formatting numbers, and the two that help format text aren't very useful.

After selecting a tag and setting a formatting option (if desired), click OK to insert the dynamic text. Dreamweaver adds a placeholder to the page; it has a blue background and displays the XPath expression inside of curly brackets, like this: {rss/channel/title}.

Tip: You can summon the XPath Expression Builder window again by double-clicking any dynamic XML text placeholder on the page.

Click an XML text placeholder to select it. You can then apply a CSS style to it, format it as a header or paragraph, or drag it to another spot on the page, just as you would any other HTML element.

Inserting a Repeat Region

XML files frequently contain the same tag repeated multiple times. For example, an XML file that's a list of employees might use the tag <employee> to begin each employee listing. For every employee in the company, the <employee> tag will appear once. The XML might look something like this:

```
<companyInfo>
<company>
<name>Big Co.</name>
<phone>555-3333</phone>
</company>
<employeeList>
<employee id="485734">
<name>Mark</name>
<phone>555-3333 x405</phone>
</employee>
<employee id="38753">
<name>Jane</name>
<phone>555-3333 x406</phone>
</employee>
</employeeList>
</companyInfo>
```

If you added the <name> tag inside the first <employee> tag to an XSLT style sheet, attached that XSL file to a dynamic page, and then previewed it in a Web browser, you would see just a single name: the first employee name in the XML file. But just as with recordsets, you usually want to display multiple XML records. The answer is Dreamweaver's XSLT Repeat Region object. To use it:

1. **Insert elements that appear within a tag that is repeated multiple times.** (**Use any of the methods described on page 926.**)

   The Bindings panel lets you know if an XML tag is repeated multiple times: check for a tiny + symbol floating just above the right side of the <> icon in the panel (see Figure 26-3).

   So, in the above employee list example, you wouldn't insert the <name> tag that appears inside the <company> tag, since <company> appears only once in the file. You would, however, insert the <name> tag and perhaps the <phone> tag inside the <employee> tag, since these are both "children" of a tag that's repeated twice in the document.

Note: This example points out a sometimes confusing aspect of XML: Tags with the same name may appear as children within different kinds of tags. The <name> tag, for instance, appears both within the <company> and <employee> tags, but obviously refers to two different things–the name of a business and the name of a person.

Articles in a Web feed are another case. For example, the RSS standard (see page 923) requires that each news item delivered in an RSS XML document be surrounded by an <item> tag with the following children: <title>, <link>, and <description>. Therefore, for an RSS feed, the elements you'd want to add to the page (and repeat once for each news item) would be <title>, <link>, and <description>.

2. **Select (by dragging, for example) the XML placeholders and any other content that you want to be repeated once for each instance in the XML file.**

   At the very least, this includes the XML placeholders you inserted in step 1, but may also include other HTML elements, such as a graphic that's repeated once for each item or a <div> tag that contains the XML data you're repeating. You can select only elements that are together: You can't, for instance, select an XML element at the top of the page and another at the bottom of the page, and use the same Repeat Region object.

Tip: You can, however, include multiple repeat regions on a page, so you could repeat the same XML data in several locations on a page by adding multiple Repeat Region objects.

3. **Choose Insert → XML Objects → Repeat Region, or click the Repeating Region button on the XSLT tab (see Figure 26-5).**

   The XPath Expression Builder window appears (see Figure 26-7). This is a similar window to the one that appears when you insert dynamic text (Figure 26-6). However, instead of a format menu, the window includes a "Build Filter" option.

4. **Select the repeating tag.**

   This tag will always have a + to the right of its <> icon, and is usually the parent tag of the tags you inserted in step 1. So, in the employee list example above, you would select the <employee> tag; in the case of an RSS feed, it would be the <item> tag.

5. **Build a filter to limit the information retrieved from the XML file.**

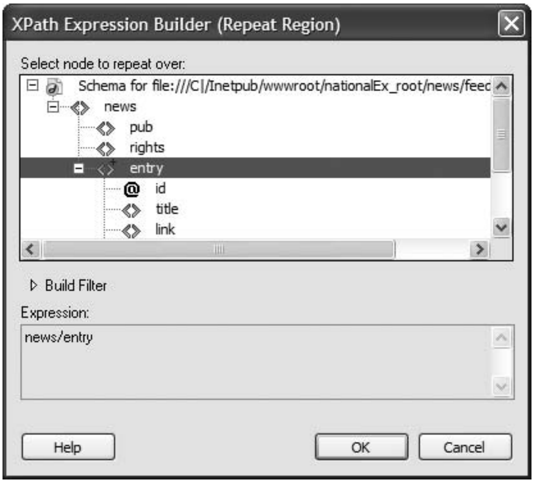
   An XSLT filter works similarly to filters on recordsets (see page 785). It's a way to select only certain information from the XML file. For example, you might want to select only employees whose last name is Smith, or product tags that have only an <instock> XML tag containing the word "true." Filters are discussed on page 932.

6. **Click OK to insert the repeat region.**

   Dreamweaver adds a gray border around the repeating elements and adds a gray tab labeled "xsl: for-each." (If you don't see these, make sure invisible elements are turned on: View → Visual Aids → Invisible Elements.)

You can see the effect by pressing F12 (Option-F12): Dreamweaver translates all of that XSLT gobbledygook into a temporary HTML file. But to see the final presentation, you need to attach the XSLT style sheet to a dynamic page (steps 8–11 on page 925) and preview it in a browser.

If you want to edit the repeat region, click the gray tab to select it and, in the Property inspector, click the lightning-bolt icon to open the Repeat Region window again (Figure 26-7).


XPath Expression Builder (Repeat Region)
Select node to repeat over:
Schema for file:///C|/Inetpub/wwwroot/nationalEx_root/news/feec
news
pub
rights
entry
id
title
link
Build Filter
Help
OK
Cancel

Expression:

news/entry

Figure 26-7:
Display repeating XML data using the XPath Expression Builder for repeat regions.

To remove a repeat region, right-click (Control-click) on the gray "xsl: for-each" tab and select "Remove Tag: <xsl:for-each>." You can also click anywhere inside the repeat region, right-click (Control-click) on "xsl: for-each" in the Tag selector (see page xx), and then choose Remove Tag. Don't try to remove the tag by hand in Code view: The code used to specify the tags inside the region also must be changed; Dreamweaver does this automatically and accurately.

Building a repeat-region filter

If the XML file you're using has lots and lots of repeating items, or you just want to hone in on a single item, you can build an XSLT filter that lets you search and select XML elements that match certain criteria. Say you want to display only employee tags with a "department" property whose value is "marketing." Fortunately, Dreamweaver lets you create very complex filters. In a nutshell, to filter a repeat region:

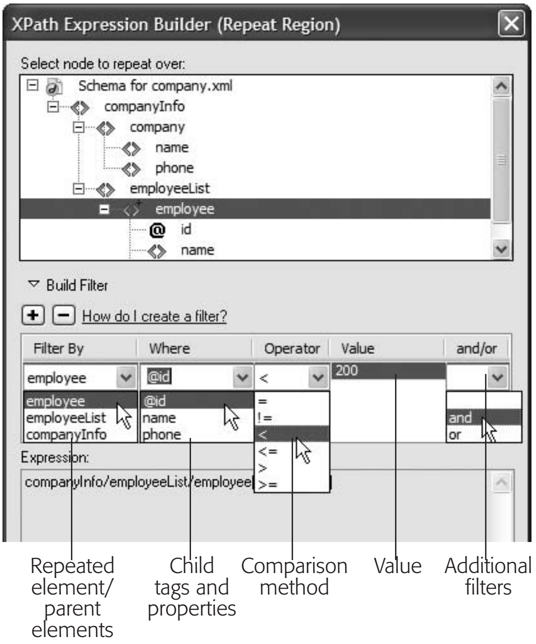
1. **Follow steps 1–4 on page 930 to insert a repeat region.**

2. **In the XPath Expression Builder (Repeat Region) window, click Build Filter to display the filter tools (see Figure 26-8), and then click the + button to add a filter.**

   You build a filter by first selecting a tag that contains the information you wish to compare to a certain value.

3. **Click in the Filter By column and, from the pop-up menu, select a tag.**

   This menu lists the repeating tag, its parent tag, its parent's parent tag, and so on, up the food chain, until it reaches the top (root) element. You'll just leave it as the repeating tag you selected in step 4 on page 931 (finish reading these steps and then read the following note to understand why this is the case).



Filter By	Where	Operator	Value	and/or
employee	@id	<	200	

Expression:

```
companyInfo/employeeList/employee
```

Figure 26-8:
Limit the XML data displayed by creating a filter that includes only XML tags that match certain criteria. Here's an example using the employee list XML code on page 930.

Note (hold onto your thinking caps): A filter lets you select criteria that each repeated region is tested against. If it passes the test, then the XML data is displayed. For example, the "id" property of the <employee> tag will vary with each employee listing. In a repeated region, the only elements that change are either a property of the tag that's repeated or the contents of other tags inside the repeated tag. That's why you should always select the repeated tag from the Filter By menu; the parent (and grandparent, and so on) of the repeated tag doesn't change with each region that repeats. If the parent has a property named "version," that property value will be the same whenever the filter is applied to a repeat region. In other words, the filter will either always be true or never be true, and you'll either get all of the XML data or none of the XML data from the repeated tags. Dreamweaver includes a more flexible tool for displaying or hiding information based on some "test" or condition: conditional regions (see page 934).

4. **Click in the Where column, and select an option from the pop-up menu.**

   This menu lists any properties of the repeated tag, and all the repeated tag's child tags. In the employee list code (page 930), for example, each <employee> tag has a property named "id" and child tags called <name> and <phone>. So in this case, those options are listed in the "Where" menu. Tag properties begin with an @ symbol, so in this example, the "id" property is listed as "@id" in the menu (see Figure 26-8).

   To continue with the employee list example, if you want to display only employees whose employee IDs are below a certain number (perhaps to list the company's first 200 employees), then choose "@id" from the menu.

Note to Power Users: If you're up on your XPath expressions (and who isn't?), you can actually click in the Where column and type your own path to identify tags and properties located deeper in the tag structure.

5. **Select a comparison method from the Operator menu.**

   Your options are = (equal to), != (not equal to), < (less than), <= (less than or equal to), > (greater than), and >= (greater than or equal to). If the property or tag you selected in step 4 contains a number, you can use any of these *comparison operators*. So if you want to find employees whose IDs are below 200, then select <.

   For properties and tags that contain text (<department>marketing</department>, for example) stick to either the = or != options. That way, a repeat region shows only employees who are either in the "marketing" department (use the = sign) or not in it (use the != operator).

6. **Type a comparison value in the Value box.**

   The value is what you're testing against. If you're looking for employee IDs that are below 200, type *200*; for <department> tags that contain the word "marketing" type *marketing*.

7. **If you want to add more filters, select either "and" or "or" from the "and/or" menu, click the + button to add another filter, and then repeat steps 4–7.**

   This lets you add additional conditions that must be met in order to select XML data to include in the repeat region. Say you want to display employees who are both in the marketing department *and* are one of the first 200 employees. In that case, select the "and" option and add another filter. Or suppose you want to display a list of employees who are *either* in the marketing department *or* the finance department: Select "or" and add a filter where the <department> tag is equal to "finance".

   The ability to add multiple filters lets you build up complex filters that either let you narrow the number of regions that are repeated (by adding more and more *and* options) or that include more and more data from the XML file (by using additional *or* filters).

8. **After adding one or more filters, click the Close button to create a filtered repeat region.**

   Dreamweaver inserts the repeat region into the XSLT style sheet. You can edit or remove this region as described on page 932.

Inserting a Conditional Region

At times, you may want to display a part of a page only if certain conditions are met. The "Filter" feature of the Repeat Region tool (see page 932) offers some help, since it can display select XML data when a tag's property or contents pass a particular test: an *id* property that's less than 200, for example. But there are other occasions when the filter doesn't help. Say you want to display only the last item in a repeat region; there's no tag or property containing this information, so a filter won't work.

Note: If you use Dreamweaver templates, this problem may sound familiar. It's the same concept as a template optional region (see page 681).

Or maybe you want to display a graphic or another part of a page only when a certain XML property appears. Suppose an XML document listing products has a tag like this: <product stock="in">. The product tag's "stock" property serves to indicate whether a product is available (in which case, it's value is "in") or when it's not ("out"). In such cases, you can use a conditional region to display an "out of stock" button next to each product that's not available.

To use a conditional region:

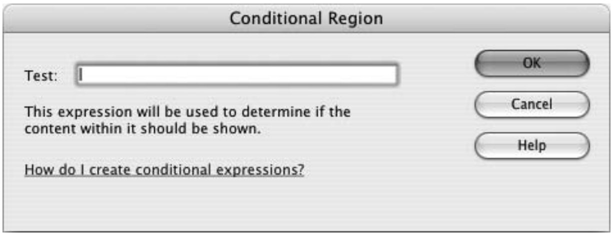
1. **Select the part of the page—either the XSLT file or XSLT fragment—you want to display if a condition is true.**

   A simple example is an "out of stock" or "on sale" graphic. But you could also select XML data placeholders: Maybe you want to display just the first five items inside a repeat region. In this case, select all the XML placeholders inside the repeat region (you need to add the repeat region first).

Note: Many Web designers find it useful to place conditional regions inside of repeat regions, since this lets them fine-tune the display of information on a per-item basis. For example, in a repeating list of products, showing an "on sale" graphic only for those products that are actually on sale.

2. **Choose Insert → XSLT Objects → Conditional Region or click the Conditional Region button in the XSLT tab (see Figure 26-5).**

   The Conditional Region window opens (see Figure 26-9).


Conditional Region

Test:

This expression will be used to determine if the content within it should be shown.

How do I create conditional expressions?

Figure 26-9:
The Conditional Region window lets you show or hide content on your page based upon certain conditions in the XML or XSL files.

3. **Type a test condition in the Test box.**

   "But what am I supposed to type?" you're asking. This is the tricky part, since Dreamweaver doesn't really give you much help. Your test condition can actually be a number of different things, many of which can be quite complex. Here are a few examples:

   - **An XPath expression followed by some kind of comparison.** For example, say you're working with the XML document on page 930. You've created a repeat region listing all of your company's employees, and you want an "employee of the month" graphic to appear in the listing, but only next to the employee whose ID is, say, 38753. To make that happen, the condition you'd type would be *@id=38753*. @id refers to the "id" property (@ is used before a property name) of the repeated tag (<employee>, in this example.) Likewise, if you wanted to highlight all employees named Jane (that is, the text inside the <name> tag is *Jane*), the condition would be *name='Jane'*. (Note that whenever you're testing whether a tag has text inside it—as opposed to just numbers—you must place quotes around the word, like this: 'Jane'.)

- **The position of an item in a repeated region.** When applying conditions to content that comes from a repeat region, you can access an item's position using position( ). So if you wanted to have the selected page elements inside a repeat region appear only when the first item is displayed, you could type *position()=first( )*; for the last item, the condition would be *position()=last( )*. And if you wanted to limit the repeat region to just 5 items (if you want to show only the first 5 headlines from a newsfeed, say), you could use this expression: *position( )<=5*.
- **An XPath expression to determine if a tag or property exists.** You can also just enter an XPath expression for a particular *node* (page 484) in the document. If the node exists, then the selected element is displayed; otherwise, it's hidden. For example, say you have a repeat region that contains some optional tags. Again, using the employee list example, imagine that some employees have their own offices. For those employees, you might add an XML tag called <office> that includes the office number, like so: <office> Room 222</office>. You'd like to include the text "Office:" followed by the actual office number in your final Web page. However, if someone doesn't have an office (meaning that her entry in the XML file has no <office> tag), you don't want the word "Office:" to appear. To make that happen with a conditional region, type *Office:* somewhere inside the repeat region (perhaps on a line below the employee phone number); next, drag the <office> tag from the Bindings panel to the page, and then select both the text and the XML placeholder. Finally, add a conditional region as described on page 934 and simply type *office* as the condition. Now "Office:" and the office number will appear only for <employee> tags that have an <office> tag inside them.
- **Tag or property values that begin with one or more particular characters.** Say you want to display only those employees whose names begin with 'M.' You can do this easily with the starts-with( ) function. In the Conditional Region box, you'd type *starts-with(name, 'M')*. Translated from XSLT-speak, this means any <name> tag whose contents start with the letter M will appear on the final Web page; so <name>Mark</name> and <name> Mary</name> would match, but <name>Andrea</name> wouldn't.

4. **Click OK to insert the conditional region.**

Dreamweaver adds a gray border around the page elements you selected in step 1 and adds a gray tab labeled "xsl:if" to indicate the conditional region. (If you don't see these, make sure invisible elements are turned on by choosing View → Visual Aids → Invisible Elements.)

You can still edit the page elements inside the conditional region's gray border: You can edit, add, or remove text, images, and XML placeholders.

If you want to edit the conditional test, click the gray "xsl:if" tab to select the conditional region, and then change the test listed in the Property inspector.

To remove a conditional region, right-click (Control-click) the gray "xsl:if" tab, and then select "Remove Tag <xsl:if>". You can also click anywhere inside the conditional region, right-click "xsl:if" in the Tag selector (see page 22), and then choose "Remove Tag".

Using Multiple Conditional Regions

A conditional region is pretty straightforward: It either shows or hides part of the page based on the results of a simple test. But what if you want to display one thing if the condition is true, but show different stuff if the condition is false? Say you have two graphics called "In Stock" and "Out of Stock" that need to appear next to each product name in a repeat region. You can use two conditional regions: the first to display the "In Stock" image if the product tag's stock property is set to "in" (<product stock="in">) and another for out-of-stock products (<product stock="out").

Note: If you've ever done any computer programming, you'll recognize the upcoming maneuver as a variation on the venerable "if-then-else" statement.

But using conditional regions in that way requires far too much work on your part. Fortunately, Dreamweaver's Multiple Conditional Region tool makes it easy to deal with these "either/or" situations. Here's how to use it:

1. **Select the part of the page you want to display if a condition is true.**

   This could be a graphical button with the text "In Stock" printed across it. This step is the same as a conditional region described on page 934. In fact, most of the steps are the same.

2. **Choose Insert → XSLT Objects → Multiple Conditional Region or click the Multiple Conditional Region button in the XSLT tab (see Figure 26-5).**

   The Multiple Conditional Region window opens. Except for its title, this window is identical to the Conditional Region window (see Figure 26-9).

3. **Type a test condition in the Test box.**

   For instance, *@stock="in"* would cause the region to display if the value of the repeating tag's *stock* property was "in." For more examples, see page 935.

4. **Click OK.**

   Dreamweaver inserts three different sections of XSL code, each marked with their own gray tab: "XSL:choose", "XSL:when", and "XSL:otherwise". The "XSL:when" section contains the actual condition or test you set in step 3.

The “XSL:otherwise” section is the part of the page that will display if the test *isn’t* true. Dreamweaver adds “Contents goes here” to that area.

5. **Select and delete “Content goes here” and then add the page elements you wish to display if the test from step 3 isn’t true.**

   This is the alternative to the content selected in step 1—for example, an “Out of Stock” icon.

You can edit the contents of either the “XSL:when” or “XSL:otherwise” sections. To edit the test, either click the gray “XSL:when” tab or click anywhere inside the “XSL:when” section and use the Tag selector (see page 22) to select the <xsl:when> tag. The Property inspector displays the test condition; edit it, and then press Enter or Return.

Removing a multiple conditional region is a bit trickier. You can’t just right-click (Control-click) the gray “XSL:choose” tab and then select “Remove Tag <xsl: choose>” to remove all of the multiple conditional region code. You must remove each of the three sections separately. To do so, follow the same process as required when removing a conditional tag, as described on page 937.

Advanced XSLT Tricks

XSLT is a complex language with lots of bells and whistles—and just as many pitfalls. It’s all too easy to head ambitiously into Code view and, with just a few keystrokes, completely break your XSLT style sheet. But since Dreamweaver’s XSLT tools take you only so far, you’ll undoubtedly find yourself wanting to dip into the code. Here are a couple of examples to help your explorations go a little more smoothly.

Sorting Data in a Repeat Region

The Repeat Region feature normally works by spitting out data that it retrieves from an XML document in the order it appears in the XML file. But what if you want that information sorted in a particular way—employees listed in alphabetical order, for example. Dreamweaver doesn’t have a visual tool to let you accomplish this common goal. Fortunately, adding the code yourself is pretty easy:

1. **Click inside a repeat region and then click the “Code” or “Split” buttons in the document window’s toolbar.**

   Alternatively, you can choose View → Code or View → “Code and Design”. Doing so drops you into the scary world of XSLT code. Don’t look too hard—you might go blind.

2. **Locate the beginning of the repeat region.**

   What you’re looking for is something like this: <xsl:for-each select="companyInfo/employeeList/employee">, where the stuff in quotes after “select” is the XPath expression pointing to the repeating tag. You need to add your new code directly after this tag.

3. **Click immediately after the closing bracket (>), hit Enter, and then type *<xsl: sort select="xml_tag_to_sort_on" data-type="text" order="ascending" />*.**

   Replace *xml_tag_to_sort_on* with the XML tag inside the repeat region that you wish to use as the basis for sorting. For example, pick a tag used for a name or a price.

Note: Don't forget the forward slash at the end of the sort tag: */>*. The tag you're adding is an *empty tag* (meaning there's no accompanying closing tag). In XML, these types of tags must be "self closed" using the forward slash (see page 479 for details).

   The value for *data-type* can be either "text" or "number." Pick the one that matches the type of data contained in the XML tag you're using as a sorting key—use "text" if you're sorting names and "number" if you're sorting prices.

   Depending on how you want to sort the data—smallest number to largest, or largest number to smallest—type either *ascending* or *descending*, respectively, for the *order* property. "Ascending" gets you smallest number to largest, or A–Z; "descending" results in largest number to smallest, or Z–A.

Using XSLT Parameters

The Repeat Region's filter feature is very useful. With it, you can winnow down a mass of XML data to a smaller collection of useful facts. But what if you wanted the data retrieved from the XML file to *change* based on information from a database or information submitted by a visitor? Say you've already created an employee list page, and now you want to create separate pages for each employee (kind of like the master-detail pages described on page 814). You could create an XSLT style sheet for each employee, thereby filtering the XML file based on the employee's ID number. But that's a lot of work. A better approach is to use an *XSLT parameter*.

XSLT parameters provide a way of passing information from an outside source to the XSLT style sheet; the parameters can affect how the XSLT style sheet processes and displays the XML file. You've already encountered one way to pass a parameter to an XSLT style sheet—the XSL Transformation server behavior (see Figure 26-2). You can use the server behavior to pass either a value you manually enter, a dynamic value pulled from a database, or any of the other sources of data accessible in dynamic Web pages (see page 897). In this way, you could present a separate page for each employee simply by passing the employee's ID number to the XSLT style sheet (instead of manually creating separate XSLT files for each employee).

For this maneuver to work, you need to string together several different concepts involving both dynamic pages and XSLT files that you've already learned in this book. Here's an example of how to use XSLT parameters to dynamically filter XML data:

1. **Create an XSLT fragment as described on page 923.**

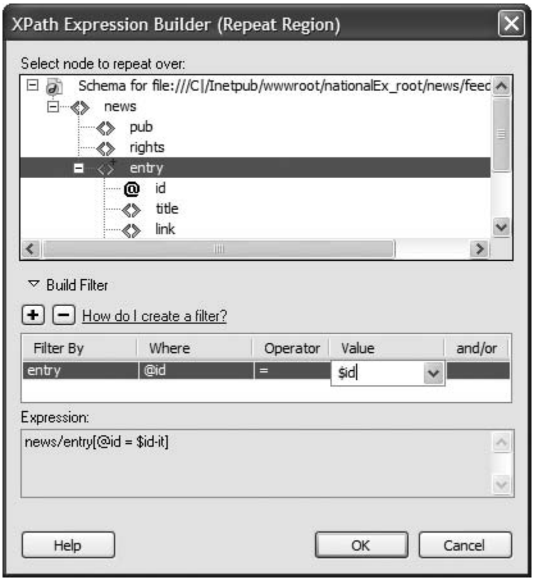
   You'll eventually include this fragment on a dynamic page (PHP, ASP, or whichever server method you're using) to display the final, filtered data.

2. **Follow steps 1–5 on page 930 to insert a repeat region and create a filter.**

   With this technique, all the steps in creating a filter are the same as those on page 932, except for entering the value in the Value box, as explained in the next step.

3. **In the filter's Value box type *$your_param* (see Figure 26-10).**

   Change *your_param* to a name you'd like to use for the parameter. For example, if you want to filter for an ID that matches a particular value, then you'd type *$id*. You must include the $ sign, but you can come up with whatever name you like. It helps if it's descriptive, like *$id, $name,* or *$price.* It also has to follow a few rules: use only numbers and letters, always start the name with a letter (not a number), don't use spaces, and stay away from punctuation marks, except for hyphens and underscores.



XPath Expression Builder (Repeat Region)
Select node to repeat over:
Schema for file:///C|/Inetpub/wwwroot/nationalEx_root/news/feec
news
pub
rights
entry
id
title
link
Build Filter
How do I create a filter?
Expression:
Help
OK
Cancel

Filter By	Where	Operator	Value	and/or
entry	@id	=	$id	

```
news/entry[@id = $id-it]
```

Figure 26-10: *You can use an XSLT parameter as a filter value. The parameter always begins with the $ sign and lets you dynamically filter the contents of an XML file.*

Unlike a static value that you type into the Value box, like *38, Dave,* or *marketing,* a parameter can be different each time the XSLT style sheet does its magic. But to get it to work, you need to dip (just a bit) into Code view.

4. **Click the "Code" or "Split" button to view the XSLT code. Locate the line** <xsl:template match="/"> **, and then click just before the opening bracket** (<).

   In XSLT, you first need to tell the style sheet that you'll be using a parameter.

5. **Type** *<xsl:param name="your_param"/>.*

   Replace *"your_param"* with the text you typed in step 3. Note that you leave off the $ sign. Also, make sure you include the forward slash before the final bracket, like this: />.

You're done with the XSLT style sheet. It's all primed to have dynamic data sent to it. The next steps involve adding the XSLT fragment to a dynamic page.

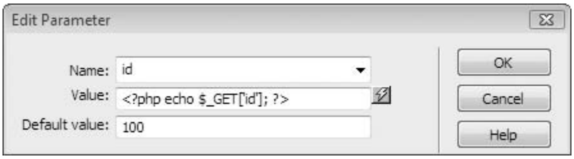
6. **Repeat steps 8–10 on page 925.**

   This step is the same process as adding any XSLT fragment to a dynamic page—that is, using the XSL Transformation server behavior.

   In the next step, you add the XSLT parameter.

7. **In the XSL Transformation window, click the + button next to the label XSLT Parameters (see Figure 26-2).**

   The Edit Parameter window opens (Figure 26-11).



Name: id

Value: <?php echo $_GET['id']; ?>

Default value: 100

Figure 26-11: *While inserting the XSLT server behavior, you can add one or more parameters that let you pass information to the XSLT fragment. That way, you can control which data from the XML file is displayed on the Web page.*

8. **In the Name box, type the name you used in steps 3 and 5 above (don't include the $ sign).**

   The value you enter here defines the name of the parameter that the dynamic page will pass off to the XSLT style sheet. Next (and this is the magic part), you'll add the value.

9. **Click the lightning-bolt icon to the right of the Value box to open the Dynamic Data window.**

   This is the same Dynamic Data window you've encountered with dynamic pages (see Figure 24-7). Don't get it confused with the XSLT Dynamic Text box (Figure 26-6). Here, "dynamic" refers to any of the many sources of dynamic information you've used when creating database-driven pages—recordsets, URL variables, form variables, cookies, session variables, and so on. (For a recap on creating recordsets, see page 782; the other types of dynamic data can be added to the Dynamic Data window as described on page 897.)

10. **Select a source from the Field list and then click OK.**

    What you select here is the crucial part of the puzzle. You're telling the dynamic page where to get the information that will be passed off to the XSLT style sheet. To use the employee list example again, you would need to identify where the ID number used to select just a single employee comes from. Here are a few examples:

    - **Recordsets.** You could use the value from a field in a recordset. For this to work, you'll need to add a recordset (pgae 782) to the dynamic page first.

- **Form fields.** One way to pass a value to a page is via a form. For example, you could add a form to a separate Web page. The form submits to this dynamic page (the one with the XSL Transformation) and includes a form menu that lists every employee's name (and includes the employee ID in the menu's value column—see page 414 for more on form menus). When a visitor selects a name from the menu, the employee ID is submitted to the dynamic page, which turns it into an XLST parameter and hands it off to the XSLT style sheet for use in the repeat region filter. (Turn to page 897 to see how to add a form field name to the dynamic data window.)
- **URL variables.** You can apply the same idea to URL variables, but instead of getting the employee ID from a menu, you'd attach it to a link to the dynamic page. For example, you might use a URL variable that looks something like this: *employee.php?id=15*. (Turn to page 897 to see how to add a URL variable to the Dynamic Data window.)

These are just a few examples. You can use dynamic data from any dynamic source: cookies, session variables, and so on.

11. **Type a value in the "Default value" box.**

    This is the value the dynamic page will use if the source you picked in step 10 doesn't come through—for example, if the dynamic page is accessed without adding a URL variable (in which case you'd be passing just *employee.php*, instead of *employee.php?id=15*). Entering a default value will ensure that the XSLT style sheet has some value to work with. If, as in this example, you're using this technique to dynamically control XML filtering, the default value should match at least one record in the XML file.

12. **Click OK to close the Edit Parameter box, and then click OK once again to close the XSL Transformation window.**

    Dreamweaver adds the new server behavior and the new parameter to your page.

Note: If you want to remove or change the XSLT parameter, just re-open the XSL Transformation window by double-clicking its name in the Server Behaviors panel.

13. **Provide a way to pass the dynamic data to the page.**

    For example, if you selected a URL variable as the data source for step 10, you would add links to other pages on your site that would point to the page with the XSL Transformation—*products.php?sku=10294* or *employee.asp?id=15*, for example.

Hopefully, by this point, your brain hasn't completely melted. As you can see, XML, XSLT, and all of the other X's can be pretty X-hausting.

7

Part Seven: Appendixes

APPENDIX
A

Getting Help

Hard as it may be to believe, even this book may not answer all your questions about Dreamweaver. Fortunately, a wide range of other resources awaits when a particular feature doesn't work for you.

Getting Help from Dreamweaver

There's plenty of assistance built right into the program, from beginner tutorials to a complete browser-based help system. You can also access Dreamweaver's electronic help system and online support center from the Help menu.

What's New

If you want a comprehensive overview of Dreamweaver CS3's new features, select the "What's New in Dreamweaver" option from the Help menu. The Help system launches and you can access a categorized list of new features that includes short descriptions and links to more detailed discussions in the electronic help system. Unfortunately, they neglected to put references to the appropriate pages in this *Missing Manual*.

Detailed Assistance

For detailed information on specific features of the program, turn to the Dreamweaver Help system. This electronic guide includes information on all the program's features. While the documentation is much better than in previous versions of the program, its coverage sometimes lacks detail. Choose Help → Dreamweaver Help or press F1 to open this help system.

Adobe also has a more interactive version of their help system online—it's called LiveDocs. This Web-based reference includes much of the same content you'll find in the program's help system, with one notable addition: the ability to leave comments. It's a great way for Dreamweaver fans to point out problems in the documentation, clarify confusing explanations, and share undocumented tips and tricks.

To get to this tool from within Dreamweaver, choose Help → Help Resources Online. This opens a browser window and connects to Adobe's Web site, so you need to be connected to the Internet. The Web page that loads includes links to "livedocs" for Dreamweaver, and, if you're the programming type, developer guides for the Spry framework, as well as two guides for extending Dreamweaver's features with your own Extensions (see page 738 for more on Extensions).

Tip: If you're the type who'd rather print out an entire manual and read it offline, you can download PDF files of the documentation on the Web page that appears when you choose Help → Help Resources Online.

If you're interested in writing your own Web code, Dreamweaver's Reference window (select Help → Reference or Window → Reference, or press Shift-F1) provides in-depth information on HTML, CSS, JavaScript, ASP, JSP, ColdFusion, and Web accessibility.

Getting Help from Adobe

You can also get more up-to-date and personalized support from Adobe, ranging from technical notes on the Adobe Web site to pay-as-you-play support plans.

Adobe Web Site

www.adobe.com/support/dreamweaver/

The Dreamweaver support page (also available from Help → Dreameaver Support Center) is your command center for finding help from Adobe. You can click the Contact Customer Service or Contact Technical Support links to create an ominous sounding "Customer Service Web Case" to request support and track your support requests. In addition, the Support Center page lists the top support issues and recent Dreameweaver-related documents that have been added by the support team. You can also search the vast database of technical notes (short articles on specific, tweaky problems) that just may hold the answer you're seeking.

For tutorials and in-depth articles on using Dreamweaver, turn to Adobe's Developer Center—choose Dreamweaver Developer Center from the Help menu. This site includes sample database applications, video tutorials, and in-depth articles. It's worth checking out frequently.

Paid Support

www.adobe.com/support/programs/dreamweaver/

If you have deep pockets, then you can also turn to three levels of personalized, fee-based support, ranging from $39 for a single incident to the whole-hog luxury of the Gold Support program (for pricing on this option you are instructed to "contact your Adobe reseller"—watch out!). For more information on these programs, go to the Web page whose address appears above. Each program has its own phone number, so read the Web site to determine the type of support (Bronze to Gold) that you need. If you have just a single nagging question, the single-incident help program gets you an Adobe technician who will work with you to resolve the issue. But at $39, make sure you've tried to answer the question yourself first using one of the free resources listed in this appendix. Customers in the U.S. and Canada should call 1-866-MYADOBE to order this service.

The Forums

Adobe provides both online forums that you get to with a browser and newsgroups that require newsgroup-reading software like the software built into Outlook Express. To get to either of these, choose Help → Adobe Online Forums or visit *www.adobe.com/cfusion/knowledgebase/index.cfm?id=tn_12606*. The forums are a terrific source of information, offering almost real-time answers on Dreamweaver and related Web design techniques. Adobe sponsors several forums and newsgroups. Of most interest to average Dreamweaver users are the General Discussion forum—for answers to basic questions—and the Application Development forum, where people discuss Dreamweaver's dynamic Web page features. If you're struggling with Dreamweaver's new Spry tools, the Dynamic HTML General Discussion forum is a good place to seek help. Odds are one of the many knowledgeable experts who always seem to be hanging around will come back with an answer, sometimes within minutes. (If you're new to forums, *www.adobe.com/support/forums/using.html* explains how to use them.)

You can access the Web forums for Dreamweaver directly on this page: *www.adobe.com/cfusion/webforums/forum/index.cfm?forumid=12*.

Help from the Real World

If Adobe doesn't have the answer, there's probably a Web site somewhere that does. In fact, you're likely to find more honest critiques of the program on some of these sites. Here are two of the best non-Adobe sites for answers.

DMX Zone

www.dmxzone.com/

The DMX Zone includes tutorials, extensions, and Adobe-related news. It also offers "Premium Content," a subscription-based service that provides more in-depth information on Dreamweaver.

Community MX

www.communitymx.com

Another subscription service (it's so hard to find free help these days). This site has lots of material for all things MX (Dreamweaver once had "MX" in its name), including Flash and other Adobe programs. It does have some free content as well, and it's updated regularly.

Help Creating Your Own Extensions

If you're excited by the possibilities of the Dreamweaver extensions (discussed in Chapter 21), you may want to have a go at creating your own. You should be well versed in JavaScript and have an interest in programming. To help you out, Dreamweaver includes a *detailed* electronic help system covering every aspect of extension development. Choose Help → Extending Dreamweaver, which launches an electronic help system. You can view its table of contents or use a search feature.

In addition, you'll need to learn Dreamweaver's advanced programming interface (API). An API lets you use JavaScript to communicate with and control Dreamweaver. It's not for the faint of heart, however; take a look by choosing Help → API Reference.

Finally, after creating your extension masterpiece, you can submit it to the Dreamweaver Exchange to show off your programming talent to the world. This Web page has more information on how to do that: *www.adobe.com/cfusion/exchange/upload/*.

APPENDIX
B

Dreamweaver CS3, Menu by Menu

Dreamweaver CS3: The Missing Manual is quite complete; in its pages, you'll find descriptions of every major Dreamweaver function (and most minor ones). In the interests of completeness, however, here's a quick reference to every command in every menu—and the answer to the occasional "What does that mean?" mystery.

File Menu

The commands in the File menu control the open Dreamweaver document as a whole. They also include basic file functions like saving and quitting:

- **New.** Opens the New Document window, which lets you select a new, blank document among many different types, from basic HTML pages to dynamic pages like ASP or PHP. This window also lets you access templates you've created for your site.
- **Open.** Opens the standard Open File dialog box so you can choose an existing Dreamweaver document to open. You can set the Show pop-up menu to show only specific types of documents—only HTML or style sheets, for example. The Preview button displays a thumbnail image of the document, if one's available.
- **Browse in Bridge.** Bridge is Adobe's file manager program. It's like the Microsoft Explorer or Mac Finder. Bridge is a way to browse, find and open documents. It's part of Adobe's graphic heritage and works best with image files—in other words, Photoshop and Illustrator files, *not* Dreamweaver documents.

- **Open Recent.** Displays a submenu that lists the 10 most recently opened documents. Selecting a document from the list opens it. The last option in this menu, "Reopen Documents on Startup," is kind of cool. If you quit Dreamweaver when any documents are still open (and this option is checked), those documents automatically reopen the next time you start up Dreamweaver.
- **Open in Frame.** Opens an existing HTML page within one frame of a frameset. To make this command available, you must first click inside a frame to select it—not just in the Frameset document. The Select HTML file dialog box opens and lets you navigate to the file you wish to insert into the frame. You can also choose to make the file's URL relative to the document or the root folder, as described in Chapter 5. (See note about frames in the box on page 163.)
- **Close.** Closes the open Dreamweaver document. If you have unsaved changes, Dreamweaver gives you the opportunity to save them.
- **Close All.** Closes *all* currently open documents. If you have unsaved changes, Dreamweaver gives you the opportunity to save them.
- **Save (Save Frameset/Save Frame).** Saves any changes you've made to your document. The Save command is dimmed if you haven't made any changes to the document since the last time you saved it.

Note: If you're working on a frames-based document, this command may say Save Frameset or Save Frame, depending on what's selected.

- **Save As (Save Frameset As/Save Frame As).** Saves a copy of the current document under a new name, closing the original version and leaving the new version onscreen. Here again, if you're working on a frames-based document, this command says either Save Frameset As or Save Frame As, depending on what's selected.
- **Save All.** Saves changes to all the open documents.
- **Save to Remote Server.** Lets you save the current file to *any* site for which you've defined a remote site. In other words, if you use Dreamweaver's FTP feature to move your files to a Web server (see Chapter 17), this option lets you directly access that Web server. In fact, it lets you access any Web server for any Web site you've defined in Dreamweaver. Because of this behavior, this option can be risky. You can accidentally save a file into the wrong Web site, or in the wrong folder of the right Web site. Therefore, it's generally better to use the Files panel and its "Put Files" button—see page 620.
- **Save as Template.** Saves the current document as a template file with the suffix .dwt. The "Save as Template" dialog box appears, so you can specify the template's file name, and indicate which site it belongs to. Dreamweaver automatically saves all template documents in a Templates folder in the selected site's folder. Templates are discussed in Chapter 19.

- **Revert.** Undoes any changes you've made to the document since the last time you saved it. Edit → Undo is often a better choice; it might take a few more steps to undo all the changes you've made, but it can actually undo changes *past* your last save. So if you're one of those gotta-save-it-every-5-seconds types, the Undo command is for you.

- **Print Code.** Prints the code (that is, what you see in Code view) of the current document.

- **Import.** Lets you import data from other sources into your Dreamweaver document, such as XML data into a Template document, HTML generated by Microsoft Word, or tabular data from a spreadsheet program like Microsoft Excel. (Use the submenu to specify which.)

- **Export.** Extracts tabular data, Cascading Style Sheet styles, or template data as XML from your Dreamweaver document, for use in other applications.

- **Convert.** Converts more modern technologies such as Cascading Style Sheets into code that's understandable by older browsers. In addition, you can convert older HTML pages into a variety of more modern formats like HTML 4.01 Strict and two forms of XHTML. Unfortunately, it's kind of hit-or-miss: this feature can't always update older files to more modern standards.

- **Preview in Browser.** Opens the current document in your Web browser. By selecting Edit Browser List, you can add new browsers to, or delete browsers from, your browser list, or specify a preferred browser.

- **Check Page.** Checks the current page for a variety of problems, such as broken links, code that's incompatible with various browsers, accessibility limitations, and invalid HTML or XML code. These same tools are available from the Results panel for checking an entire site's worth of files—choose Window → Results, and then click an appropriate tab—like the Link Checker to check links.

- **Compare with Remote/Compare with Testing.** Lets you use a third-party code-comparison tool to see how a local copy of a page differs from either the remote copy (on the Web server) or a copy on your testing server. This lets you see exactly what code differs between two copies of the same page. This feature is discussed on page 387.

- **Design Notes.** Opens the Design Notes window (Chapter 17), where you can add additional information about the document, set the status, and choose to have the Design Note appear whenever the document is opened.

Note: To use Design Notes on your site, you must make sure that in the Site Definition window's "Design notes" section, the Maintain Design Notes option is selected.

- **Exit/Quit.** Exits Dreamweaver. If any of your open Dreamweaver documents have unsaved changes, the program prompts you to save them before quitting.

Edit Menu

The Edit menu applies common document changes like copying and pasting:

- **Undo.** Undoes the most recent change made to your document. You can choose this command repeatedly to move progressively backwards through your changes, even *after* you've saved the document.

- **Redo (Repeat).** Restores whatever changes you just made by using the Undo command. Selecting Redo multiple times moves you progressively forward through changes you've undone. If you've just performed an operation other than Undo, Repeat appears instead of Redo. This property lets you repeat the last action. For example, if you just pressed Delete, the Repeat command presses it again.

- **Cut.** Deletes the selected text or objects from the document, and copies them to the invisible Macintosh or Windows Clipboard so they can be pasted elsewhere. (The Clipboard holds only one selection at a time.)

- **Copy.** Copies the selected text or object to the Clipboard so it can be pasted elsewhere—without disturbing the original.

- **Paste.** Places the most recent selection from the Clipboard into your document at the insertion point.

- **Paste Special.** Opens the Paste Special window, which lets you choose how you wish to paste the Clipboard into your document. Options range from Text Only for just plain text to increasingly more elaborate options, which force Dreamweaver to attempt to preserve various levels of formatting, such as styles, bold, italics, bulleted lists, and so on.

- **Clear.** Deletes the selected text or objects from the document without placing it on the Clipboard.

- **Select All.** Selects everything in the document so you can make document-wide changes in one fell swoop.

- **Select Parent Tag.** Increases the current selection to include everything within the *parent tag*, including its content. If you had a table cell selected, this command would increase the selection to the entire table *row*. Choosing the command a second time would increase the selection to include the entire table. In short, this command ensures that any changes you make apply to the entire tag.

- **Select Child.** Decreases the current selection to include everything within the child tag, including its contents. If you selected a table row, choosing this command would decrease that selection to include only the first table *cell* and its contents.

- **Find and Replace.** Opens the "Find and Replace" window, which you can use to search the document—or entire site—for a specific word, tag, or source code, and replace it with something different (see page 718). This command lets you make such changes either en masse or one instance at a time.

- **Find Selection.** This command lets you find another instance of the current selection. Say you've selected the word "Mothball" on the page. With this command, you search the page for another example of "Mothball."

- **Find Next.** Uses the most recent search settings from the "Find and Replace" window to search the current document, highlighting the next instance of the requested search item.

- **Go to Line.** Opens the Go To Line dialog box. Type a number, and Dreamweaver positions the cursor at the beginning of the specified line of code. (Available only in Code view.)

- **Show Code Hints.** Immediately displays any code hints (overriding the delay set in the Preferences window) available for the current tag. Code Hints, described in Chapter 10, provide a pop-up menu of tag properties appropriate for the current tag (available only in Code view, and when using the Insert Tag command [Ctrl-T]).

- **Refresh Code Hints.** Doesn't seem to do much of anything.

- **Code Hint Tools.** When working in Code view, lets you access Dreamweaver's color picker, "Browse for File" button, and list of fonts so you don't have to type things like *#FF6633, ../../images/dog.gif*, or *Arial, Helvetica, sans-serif*, every time you use a color, link to a file, or want to use a font.

- **Indent Code.** Adds one indent before the selected line of code. (Available only in Code view.)

- **Outdent Code.** Removes one indent from the selected line of code. (Available only in Code view.)

- **Balance Braces.** When you're editing a script in Code view, this command helps you check for unbalanced braces (that is, an introductory "{" without a closing "}") by highlighting the matching tags enclosing the selected code. It doesn't do anything for plain HTML, but if you're writing a JavaScript program or using a dynamic programming language like PHP or ASP, it can help identify missing braces—a common source of programming errors. Works with ( ) as well.

- **Repeating Entries.** Lets you cut, copy, paste, and delete repeating regions in templates. Repeating regions are described in Chapter 19.

- **Code collapse.** Hides a selection of code in Code view, so you need to see only the code you're interested in working on. This feature is discussed on page 379, and since the same options are available more directly from the coding toolbar, you can skip these menu options.

- **Edit with External Editor.** If you haven't already specified an external HTML code editor, such as BBEdit or Notepad, to use when editing large amounts of source code, this command opens the Preferences window so that you can find and select one on your hard drive. Once you've specified an editor, this command opens the current document in that editor.

- **Tag Libraries.** Lets you modify the way Dreamweaver writes code for various types of tags: HTML, ColdFusion, ASP, and so on. You can create new tag libraries for working with other types of tag-based languages, or modify the ones that ship with Dreamweaver.
- **Keyboard Shortcuts.** Opens the Keyboard Shortcuts window, and shows you all Dreamweaver's current keyboard shortcuts. You can create a new set of shortcuts for specific sites or programs, or export the settings to HTML to share with others. (You must duplicate the factory settings before you can add or delete your own shortcuts.) Details are in Chapter 21. (In Mac OS X, this option appears under the Dreamweaver menu.)
- **Preferences.** Opens the Preference window, which is full of options that customize the way Dreamweaver works. There are 20 categories of preferences, including the color and format of different HTML tags, shorthand for CSS styles, and the order in which panels appear on the screen. (In Mac OS X, this option appears under the Dreamweaver menu.)

View Menu

The View menu controls the document window's appearance. A checkmark in the menu lets you know which view you're in:

- **Zoom In.** Zooms in on the document in 50 percent increments. If you're looking at a document at normal size (100 percent), selecting this option zooms in to 150 percent; selecting it again zooms in to 200 percent.
- **Zoom Out.** Zooms out from the document in 50 percent increments.
- **Magnification.** Lets you choose from a list of magnification levels from the absurdly small and illegible 6 percent all the way to a ridiculously large, land-of-the-giant-pixels 6,400 percent.
- **Fit Selection, Fit All, Fit Width.** Additional magnification options that zoom in or out, depending of the size of the document or selected element.
- **Code.** Displays the file's source code.
- **Design.** Displays the file's visual design.
- **Code and Design.** Splits the document window into two panes: source code on top, visual design at the bottom. You can adjust how much of each pane is visible by dragging the center divider up or down.
- **Switch Views.** Switches between the Code and Design views.
- **Refresh Design View.** Updates the Design view to reflect changes you've made directly to the source code in either Code view or split (Code and Design) view.
- **Head Content.** Opens a new menu bar in the main document window that contains shortcuts to accessing the file's Head contents. You can use these menu items to highlight your document's Title tags, meta tags, and scripts, and then, in the Property inspector, edit their content.

- **Noscript Content.** When inserting JavaScript code into the document window, you can also include what's called "Noscript" tags—information that appears in browsers that don't understand JavaScript (of which there are very few), or which have their JavaScript turned off. After selecting this option, all information inside noscript tags appears in the document window. To hide this information, select this menu option again.

- **Table Mode.** Lets you switch between the standard Table view, Expanded Tables view and something called Layout Table view. Layout Table view is a holdover from earlier versions of Dreamweaver. This view is intended to make creating table-based layouts easier, but more often creates hard-to-edit HTML. Layout Table View used to appear front and center in the program, but the Adobe engineers have hidden it away in this menu, so that those who used the tool in the past can continue to use it. But don't you be tempted to use it! CSS is a far superior way of laying out Web pages (see Chapter 9 for the details).

- **Visual Aids.** Lets you summon onscreen symbols that represent typically invisible page elements like image maps, anchors, and borders.

- **Style Rendering.** Lets you hide or show the effect of all style sheets on a page, or selectively display the formatting changes caused by a style sheet that's applied for a particular media—for example, screen only or printer only.

- **Code View Options.** Lets you adjust the appearance of your HTML code in Code view. You can turn on (or off) options that wrap lines of text to fit in the document window, add line numbers, highlight invalid HTML, turn on syntax coloring, or indent lines of code.

- **Rulers.** When you choose Show, Dreamweaver displays rulers along the top and left sides of the document window. Using the options you find here, you can choose your ruler units: pixels, inches, or centimeters. You can also reset the orientation of the two rulers so that both start from zero in the screen's upper-left corner.

- **Grid.** Places a grid of vertical and horizontal lines over the document window to use as a guide when building your layouts. Selecting Edit Grid opens the Grid Setting dialog box, where you can adjust your grid's colors, spacing, behaviors, and line appearance.

- **Guides.** Shows, hides, locks, and erases user-added guidelines that have been dragged from a ruler onto the page. Also controls options for guides, and displays guidelines that mark the visible area of a Web browser window on monitors of different resolutions.

- **Tracing Image.** Adjusts the document's background tracing image. You can load a new tracing image, make a current one visible, or adjust its position.

- **Plugins.** Lets you "play" browser plug-ins within the document window to test embedded media. You can choose to play a document's plug-ins one at a time, or all at once, to simulate how the page will look to your viewers.

- **Display External Files.** You can insert images and other files from your own or other Web sites on the Internet. When you insert an image, for example, instead of selecting a file from your site, you can type or paste an absolute URL (page 154) to a graphic located on the Internet. Dreamweaver even displays the image in Design view, but only if this option is checked. Because it depends on an Internet connection to display the image, pages with links to external files may take longer to display in Dreamweaver (since it has to get the images and files over the Web). If you have lots of external images and files, and your pages open sluggishly in Dreamweaver, uncheck this option.
- **Hide Panels (Show Panels).** Hides all open panels. If panels are already hidden, the command says Show Panels instead, and restores the panels to their original positions.
- **Toolbars.** Displays toolbars for use with Dreamweaver. Select Document from the submenu to display the Toolbar menu at the top of the document window. This menu offers common commands like the document's View settings, page title, file management options, code navigation options, and browser preview. The Standard toolbar option displays a toolbar with common buttons for common commands, such as opening files; closing files; and cutting, copying, and pasting content. The Style Rendering toolbar lets you toggle style sheets off and on like the Style Rendering menu described earlier in this section.

Insert Menu

The Insert menu adds selected page elements to the document at the insertion point's position. The commands listed here correspond to the buttons on the Objects panel:

- **Tag.** Opens the Tag Chooser window, which provides access to all the tags—not only HTML, but any tag Dreamweaver has stored in its Tag Library (see entry under the Edit menu on page 954). You can insert any tag and set any of its properties from this window. However, Dreamweaver doesn't make sure you're inserting the tag correctly, so you should understand HTML (or the tag language you're using) before trying this option.
- **Image.** Inserts an image file, such as a JPEG, PNG or GIF, into the document. The Select Image Source window appears, and lets you navigate to the file you want on your hard drive. You can choose to make the URL for the file relative to either the document or the Site Root.
- **Image Objects.** Lets you insert placeholder graphics, rollover images, or HTML from Fireworks. These options are discussed in Chapter 6. Avoid the Navigation Bar listed here—it's left over from earlier versions of Dreamweaver and it's just plain bad. You're much better off using the new Spry Navigation Bar discussed in Chapter 5.

- **Media.** Inserts other types of media files, including Flash, Shockwave, Generator Applets, Plug-ins, and Active X. In most cases, the standard Select File window appears, which you can use to navigate to the desired file. This menu also lets you insert Flash text, Flash buttons, and the Image Viewer Flash element (an old and clunky Flash-based slideshow tool).

- **Table.** Inserts a new table into the document. The Insert Table dialog box appears, and lets you format the table by specifying the number of rows and columns; the table width; measurements for cell padding, cell spacing, and the table border; and whether or not and where to include table headers.

- **Table Objects.** Provides methods to insert tabular data (see the Import entry under the File menu on page 951) and add other table-related tags such as the <th>—table header—tag into the page. The tag options listed under this menu item assume you understand HTML and let you just insert the tags, without making sure you're doing it correctly.

- **Layout Object.** Lets you insert absolutely positioned divs, regular divs, table layout cells, and table layout tables. This menu also includes the new Spry Widgets like the Spry Navigation bar discussed in Chapter 5, and the Spry panel widgets discussed in Chapter 12.

- **Form.** Inserts Form Objects—the <form> tag, text fields, buttons, checkboxes, or lists—into the document. (If you haven't already inserted the <form> tag, Dreamweaver prompts you to do so.)

- **Hyperlink.** Inserts a link. The Insert Hyperlink dialog box lets you specify the text that should appear inside the link, the file to link to, as well as many other link options such as target and tab index.

- **Email Link.** Creates a new email link at the insertion point. The Insert Email Link dialog box appears; specify both the email address and the link's text (such as "Click to email me").

- **Named Anchor.** Inserts a named anchor for adding links *within* a page. See page 167.

- **Date.** Inserts the current date into the document. The Insert Date dialog box lets you format the appearance of the day of the week, the date, and the time. You can also elect to have the date automatically updated each time the document is saved.

- **Server-Side Include.** Opens a Find File window, from which you select a file that's dynamically added to the content of your page. Works only with special server setup, such as the dynamic server-driven pages discussed in Part 6 of this book.

- **Comment.** Inserts an HTML comment into your page. This comment isn't viewable in Web browsers, but in Dreamweaver's Design view, it appears as a little gold shield. Use these to leave notes for yourself and others about specific information about a page. For example, a comment indicating where an ad should be placed can help someone updating the page.

- **HTML.** Menu including lots of specific HTML tags, such as a horizontal rule, frames, text objects (many of which are also available under the Text menu), script objects for JavaScript, and head tags that go in the *head* portion of a Web page—including meta tags such as keywords and descriptions used by some search engines.
- **Template Objects.** When working on a template file, this menu option lets you insert many of Dreamweaver's template features, such as Optional, Editable, and Repeating Regions.
- **Recent Snippets.** Lists the most recently inserted snippets. Selecting a snippet from the list inserts it into the document. Snippets are discussed in Chapter 18.
- **Spry.** Inserts any of the new Spry objects introduced in Dreamweaver CS3, including the Spry Navigation bar (Chapter 5), Spry Form Validation Widgets (Chapter 11), and Spry Data and Layout widgets (Chapter 12).
- **XSLT Objects** (visible only when working on an XSL file). Inserts various objects for converting XML data into a Web browser–readable format. This feature is discussed in Chapter 26.
- **Customize Favorites.** Lets you add your favorite objects from the Insert panel into a special "favorites" tag, so your most common objects, images, divs, rollovers, tables, can be just one click away. See page 170 for more information.
- **Get More Objects.** Opens the Adobe Exchange for Dreamweaver Web site in your browser. There you can search for and download new extensions and objects to add new features to your copy of Dreamweaver. Use the Commands → Manage Extensions command to add downloaded extensions to Dreamweaver.

Modify Menu

You can use the commands in the Modify menu to adjust the properties of common document objects: like links, tables, and layers:

- **Page Properties.** Opens the Page Properties window, where you can specify document-wide attributes—such as the page title, background and link colors, page margins, and background image—or select a *tracing image* to use as a reference for designing the page.
- **Template Properties.** Opens the Template Properties window, where you can modify settings for various template features, such as controlling the visibility of optional regions, the properties of editable attributes, and the values of any template expressions you've created. Available only when you're working on a template-based page, as described in Chapter 19.
- **Selection Properties.** When this item is selected (as indicated by a checkmark in the menu), the Properties inspector palette is on the screen; you use it to edit the current settings for selected page elements. This command is the same as choosing Window → Properties.

- **CSS Styles.** Controls the display of the CSS Styles Panel. A checkmark tells you that the panel is open. This item has the same effect as choosing CSS Styles from the Window menu.

- **Edit Tag.** Opens a dialog box with detailed options for the HTML tag that's active in the current document. This advanced feature is for the true HTML geek—it gives access to *all* the properties for a specific tag (not just the ones Dreamweaver displays in the Property inspector). But skip this option: The Tag inspector, which provides a less intrusive panel with all the same options, is better. Choose Window → Tag Inspector to open it.

- **Quick Tag Editor.** Lets you edit an HTML tag without leaving Design view. If nothing on the page is selected, the Quick Tag editor prompts you to enter a new HTML tag at the insertion point (by choosing from the alphabetical menu). But if text or an object is already selected when the Quick Tag Editor is opened, the window displays the selection's HTML tags for editing.

- **Make Link.** Turns a highlighted page element (graphic or text) into a link. The standard Select File dialog box appears; choose the document you want to open when someone clicks the link.

- **Remove Link.** This command is available only when a link is selected or the insertion point is inside a link. Remove Link deletes hyperlinks by removing the <a href> tag from the selected text or image.

- **Open Linked Page.** Opens the linked page in a new document window. This command is available only when a link is selected or the insertion point is inside a link. (You can, however, hold down the Ctrl key [⌘] and double-click a link to open the page to which it's linked.)

- **Link Target.** Sets a link's target and defines whether the linked page appears in the same browser window or a new one. You can choose from blank, parent, self, or top targets, or manually define the target in the Set Target dialog box. This command is available only when a link is selected or the insertion point is inside a link. (See Chapter 5 for details on links.)

- **Table.** Opens a list of options for modifying a selected table. You can adjust the number of rows and columns, add row or column spans, or completely clear cells' defined heights and widths (see Chapter 7).

- **Image.** Opens a list of options for modifying a selected image, including optimizing it in Fireworks, or using one of the new built-in image-editing tools, such as the crop, resample, and sharpen tools. See page 219.

- **Frameset.** Offers options for splitting the current page into *frames*. Or choose the Edit No Frames Content command to create alternative Web-page material that can be read by older browsers that don't support frames. Frames aren't used much any more, and they're best avoided by the professional Web designer.

- **Arrange.** Lets you change the Z-index (the front-to-back order) of overlapping absolutely positioned elements. You can choose to send one element in front of another absolutely positioned element, send it to the back, and so on. You can also tell Dreamweaver to prevent overlapping elements altogether. If two or more layers are selected, you can also choose from one of this menu's alignment options to align things like the tops of two elements. See Chapter 9 for more on absolutely positioned elements.

- **Convert.** Don't use this menu! It's meant to take a table-based layout and turn it into a layout using CSS absolute positioning. It doesn't work well at all. Better to recreate your design using the CSS layout techniques described in Chapter 9. The other option—converting absolutely positioned elements to table layout—produces awful HTML and no benefit (unless you're building a "Retro Web Design Circa 1998" Web site).

- **Navigation Bar.** Skip this option. It's meant to help edit the navigation bar available from the Insert → Image Objects (see "Image Objects" on page 956). The Spry Navigation bar discussed in Chapter 5 is far superior.

- **Library.** Lets you add selected document objects to the site's Library file (Chapter 18). You can also update the current document, or multiple documents, to reflect any changes you've made to a Library object.

- **Templates.** These commands affect *template* documents (Chapter 19). Using these commands, you can apply a pre-existing template to the current page, separate the page from its template, or update the page to reflect changes made to its template. If the open document is a template file, you can use this menu to create or delete editable regions (remove template markup), and update all site files based on that template. You can also use this menu to add repeating template regions and editable tag attributes.

- **Timeline.** The submenu provides options for adding or deleting timelines, animation frames, objects, or behaviors. This feature is a bit archaic and adds lots of JavaScript code to your page. If you're still interested, you can download a chapter from an earlier edition of this book, which discusses how to use this feature: *www.sawmac.com/missing/dwmx2004/DWmx_Ch12.pdf*.

Text Menu

As you can guess, the commands in the Text menu format and modify the document's text:

- **Indent.** Adds one level of indentation to everything within the current block-level element (paragraph, headline, bulleted list).

- **Outdent.** Removes one level of indentation from everything within the current block-level element.

- **Paragraph Format.** Applies a paragraph format, such as Heading 1, Heading 2, or preformatted text, to all the text in the current block-level element. You can also go to the submenu, and choose None to remove the paragraph formatting.

- **Align.** Aligns text in the selected paragraph to the left margin, center, or right margin of the document. If the paragraph is inside a table cell or layer, Dreamweaver aligns it with the left, center, or right of that cell or layer.

- **List.** Turns the selected paragraph into an ordered, unordered, or definition *list*. You can edit the list's format by selecting the submenu's Properties option.

- **Font.** Lets you choose from a list of common font combinations for application to the selected text. When displaying text, your visitor's browser moves down the list of assigned paragraph fonts until it finds one installed on its system (Chapter 3). You can create your own combination of paragraph fonts by going to the submenu, and choosing Edit Font List.

- **Style.** Applies predefined text styles—such as Bold, Italic, or Strikethrough—to the selected text.

- **CSS Styles.** Lets you create new CSS (Cascading Style Sheet) styles, and then apply them to selected text (Chapter 4). You can also choose to attach an existing style sheet to the current document, or export the document's own style sheet for use in other sites.

- **Size.** Applies a new size to the selected text. Sizes range from 1 (the smallest) to 7 (the largest); as described in Chapter 3, HTML sizes are relative, and they change depending on your visitors' browser preferences. CSS offers a much better alternative that's not only more flexible, but also uses less code and is more in line with current Web standards and techniques (see Chapter 4).

- **Size Change.** Increases or decreases the selected text's size relative to the document's base font size (which is set to 3 by default). The note about CSS in the previous item applies here, too.

- **Color.** Opens the standard Mac or Windows color picker dialog box, so that you can choose a color to apply to the selected text. *Macintosh*: You can choose from a variety of color palettes, including CMYK, RGB, HTML (Web safe), HSV, and HLS. *Windows*: In general, the Property inspector's color box is a better way to assign Web colors to text.

- **Check Spelling.** Checks the current document for possible spelling errors (see page 79).

Commands Menu

You can use the Commands menu to apply advanced features to your Dreamweaver document. Some menu items, such as the Record commands, eliminate repetitive tasks; others, such as the Clean Up HTML command, fix common problems in a single sweep:

- **Start/Stop Recording.** Records a series of actions that can then be reapplied to other parts of the document (Chapter 20). When you select the Start Recording command, Dreamweaver records each of your actions until you choose Stop Recording. Note that Dreamweaver retains only one recorded command at a time.

- **Play Recorded Command.** Reapplies the most recently recorded command.
- **Edit Command List.** Opens a list of all saved commands. You can rename the commands, or delete them permanently.
- **Get More Commands.** Opens the Adobe Exchange for Dreamweaver Web site in a new browser window so that you can search for and download new extensions or commands. Extensions are downloaded to your Extension Manager (see Chapter 21).
- **Manage Extensions.** Opens the Extension Manager, a program that lets you manage extensions you download from the Adobe Exchange Web site (Chapter 21). The Extension Manager helps you install, delete, and selectively disable extensions.
- **Apply Source Formatting.** Changes you make to Dreamweaver's HTML source formatting (which is defined in the Preferences window and the SourceFormat.txt file) apply only to newly created documents. This command, on the other hand, offers a way to apply these formatting preferences to existing HTML documents.
- **Apply Source Formatting to Selection.** Same as the previous command, "Apply Source Formatting," but applies only to whatever you've selected. In this way, you can make sure the HTML for a <table> is nicely formatted (by selecting it and applying this command), while the rest of your finely crafted HTML is left alone.
- **Clean Up HTML/XHTML.** Opens a list of options for correcting common HTML problems, such as empty tags or redundant nested tags. Once you've selected what you'd like to fix, Dreamweaver applies those changes to the current document, and, if requested, provides a log of the number and type of changes made. (See Chapter 16.)
- **Clean Up Word HTML.** If you import HTML that was generated by Microsoft Word, you often end up with unnecessary or cluttered HTML tags that can affect your site's performance. This command opens a list of options that can correct common formatting problems in Microsoft Word's HTML. Dreamweaver applies your selected changes to the document and, if requested, displays a log of the number and type of changes it made.
- **Add/Remove Netscape Resize Fix.** This command lets you insert JavaScript code into your document that counteracts a bug in version 4 of Netscape Navigator (the bug causes pages that use layers to display incorrectly when the browser window is resized). The inserted code makes the page reload every time someone resizes a browser window. Does anyone still use Netscape 4?
- **Remove Flash Video Detection.** If you've added Flash Video to your page as described on page 546, Dreamweaver inserts some JavaScript code to help make sure your site's visitors can view the video. Unfortunately, if you just delete the movie from your page, the JavaScript code is left in the page. This command removes it.

- **Optimize Image.** Opens the selected image in the Image Preview window, where you can experiment with different compression settings to find the best balance between file size and image quality. See page 205.
- **Create Web Photo Album.** Lets you turn a folder of images into a Web-based photo album. The Create Web Photo Album window appears; specify a title for your album, the source folder, and so on.

Note: This command requires Adobe's Fireworks image-editing program, which creates thumbnail and full-size versions of each image. Dreamweaver then creates a Web site with one page displaying all the thumbnail images. The thumbnails are linked to individual HTML pages containing the full-size images.

- **Sort Table.** Sorts the information in a selected table. You can choose to sort alphabetically or numerically, in ascending or descending order. You can't apply this command to tables that include *rowspans* or *colspans*.
- **Insert Mark of the Web.** This is applicable only to Windows XP with Service Pack 2 and Vista. The Service Pack 2 update for XP inserted code into Internet Explorer to "protect" it from malicious Web page code. Unfortunately, this also has the effect of preventing you from previewing JavaScript effects—like the image rollovers discussed in Chapter 6—or Flash movies. Strangely, this happens only when you preview a local page (on your own computer), not when you view a page on the Internet. This menu option lets you overcome that peculiar problem.
- **Attach an XSLT Stylesheet.** This menu option, available only when working on a XML file, lets you attach an XSL file, which miraculously transforms cryptic XML into a beautiful, browser-viewable page. This feature is discussed in Chapter 26.

Site Menu

As its name suggests, the commands in this menu apply to your entire Web site, rather than one document at a time. These commands can help keep your Web site organized, and promote collaboration between large workgroups:

- **New Site.** Opens the New Site window, where you can set up a site for working in Dreamweaver.
- **Manage Sites.** Opens the Manage Sites Panel where you can create, delete, or edit site definitions. See Chapter 15.

Note: The next five menu commands let you transfer files between your computer (the *local* site) and a Web server (the *remote* site). These commands, in other words, don't work unless you've first defined a remote site in the Site Definition window. In addition, these operations work only on files that you've *selected* in the Site window.

- **Get.** Copies files (those you've selected in the Site window) from the remote server to the local site folder for editing. Note that if the File Check In and Check Out feature is active, the downloaded files aren't editable.
- **Check Out.** Copies files (those you've selected in the Site window) from the remote server to your local site, and marks them on the remote server as *checked out*. No one else can make changes to the document until you upload it back onto the remote server.
- **Put.** Uploads files (those you've selected in the Site window) from the local site to the remote site. The uploaded file replaces the previous version of the document.
- **Check In.** Uploads checked-out files from the local site to the remote site, and makes them available for others to edit. Once a file is checked in, the version on your local site becomes read-only (you can open it, but you can't edit it).
- **Undo Check Out.** Removes the checked-out status of selected files. The file isn't uploaded back to the remote server, so any changes you made to the file aren't transmitted to the Web server. Your local copy of the file becomes read-only.
- **Show Checked Out By.** Lets you see who's checked out a particular file.
- **Locate in Site.** When working on a document, selecting this option opens the Site window and highlights that document's file in the site's local folder.

Note: See Chapter 17 for the full scoop on remote sites, local sites, and checking files in and out.

- **Reports.** Opens the Reports window, and lists options for generating new reports (Chapter 16). Reports can monitor workflow (such as design notes and checkout status) and common HTML problems (such as missing Alt text, empty tags, untitled documents, and redundant nested tags). You can generate a report on just the open document, multiple documents, or the entire site.
- **Synchronize Sitewide.** Opens the Synchronization window, which lets you compare all your local files with all the files on your Web server. Use this to make sure all the most recent files you've updated locally are transferred to the Web server, or vice versa.
- **Check Links Sitewide.** Analyzes the current site for broken links, external links, and orphaned pages. Dreamweaver then generates a report listing all the found problems. You can fix problematic links directly in the Report window—or click the file name to open the errant file in a new document window, with the link highlighted and ready to repair.
- **Change Link Sitewide.** In one step, replaces a broken link that appears multiple times throughout your site. In the Change Link dialog box, you first specify the incorrect link; below it, enter the link with which you'd like to replace it. Dreamweaver searches your site, replacing every instance of the old link.

- **Advanced.** Provides access to advanced site options, such as the FTP Log, which opens the *FTP log*—a record of all FTP file transfer activity; "Recreate Site Cache," which forces Dreamweaver to rescan the site's files and update its cache file to reflect any changes to the files or links in the site; "Remove Connection Scripts" for removing the script files Dreamweaver creates to work with dynamic, database-driven Web sites; and "Deploy Supporting Files" to move necessary programming files to the Web server when using Dreamweaver's ASP.NET server model to build dynamic pages.

Window Menu

This menu controls which panels and windows are visible or hidden at the moment. (A checkmark in the menu denotes open panels.)

- **Insert.** Opens the Insert bar, from which you can insert various types of objects (such as images, layers, or forms) into your document. The Insert bar also contains options for switching between Layout and Standard table views, and accessing options for dynamic Web pages.

- **Properties.** Opens the Property inspector, where you can edit the relevant properties for a selected object. The options in the Property inspector depend on which page element's selected.

- **CSS Styles.** Opens the CSS (Cascading Style Sheet) Styles panel, from which you can define and edit CSS styles, or apply existing ones to selected text.

- **AP Elements.** Opens the AP Elements panel, which lists all elements that have been positioned on the page using CSS positioning. See Chapter 9 for more details.

- **Databases.** Opens the Databases panel for working with dynamic Web sites. This panel lets you connect your site to a database, view the structure of a database, and even preview data currently stored in the database.

- **Bindings.** Opens the Bindings panel, which lets you create database queries for working with dynamic sites. In addition, the panel displays and lets you add dynamic data to a Web page.

- **Server Behaviors.** Opens the Server Behaviors panel, the control panel for viewing, editing, and adding advanced functionality to dynamic Web pages.

- **Components.** Opens the Components panel, for use with ColdFusion MX and JSP sites, as well as Web Services. This advanced feature lets ColdFusion and JSP developers take advantage of prewritten, self-contained programs, which makes building complex dynamic sites easier.

- **Files.** Opens the Files panel. From this window, you can open any file, and transfer files between your computer and the remote server.

- **Assets.** Opens the Assets panel, which conveniently groups and lists all the assets (such as colors, links, scripts, or graphics) you've used in your site.

- **Snippets.** Opens the Snippets panel, which contains snippets of HTML, JavaScript, and other types of programming code. You can create your own snippets to save your fingers from having to retype frequently repeated code.

- **Tag Inspector.** Opens the Tag Inspector panel, which provides a listing of *all* properties available for the currently selected HTML tag. This uber-geek option is like the Property inspector on steroids.

- **Behaviors.** Opens the Behaviors panel, which lets you associate *behaviors* (such as swapping images in a rollover, or checking for necessary plug-ins) to selected page elements (see Chapter 13).

- **Results.** Lets you open Dreamweaver's many site tools, such as the Find and Replace command, Link Checker, and Reports command. Pick the type of site-wide action you'd like to perform using the submenu.

- **Reference.** Opens the Reference panel, a searchable guide to HTML tags, Cascading Style Sheets, and JavaScript commands. The guides are culled from the popular O'Reilly reference books and include an explanation of what specific tags do, when you can use them, and what additional components are required, as well as tips for getting the most out of them.

- **History.** Displays the History panel for viewing a record of actions performed in the current document.

- **Frames.** Displays the Frames panel for selecting frames and framesets for editing.

- **Code Inspector.** A window displaying the HTML code for the current document. You can edit the code directly in the window, while still looking at the Design view. It's often easier to just use Dreamweaver's "Code and Design" view (View → Code and Design).

- **Timelines.** Opens the Timelines panel, in which you can set up and refine animations within Dreamweaver. This feature was added in Dreamweaver MX, and then removed in Dreamweaver MX 2004…and *then* put *back in* with the Dreamweaver 7.01 updater. This feature is a bit archaic, and adds lots of JavaScript code to your page, so this edition doesn't include details on how to use it. You can, however, download the chapter from an earlier edition of this book that describes how timelines work: *www.sawmac.com/missing/dwmx2004/DWmx_Ch12.pdf*.

- **Workspace Layout.** Lets you save the position and size of Dreamweaver's panels and windows in any arrangement you like.

- **Hide Panels.** Closes all currently open panels. Choosing Show Panels reopens only those panels that were displayed before you selected Hide Panels.

- **Cascade.** By default, when there are multiple documents open, you switch from page to page by clicking on tabs that appear at the top of the document area. If you prefer to have all open documents floating and resizable within this space, this and the next two options let you "undock" the current documents. The cascade option resizes each open document and places them one on top of the other. Windows folks can redock pages by clicking the Maximize button on any currently opened document. Mac people can select the Combine As Tabs option.

- **Tile Horizontally (Windows Only).** Places all open documents one on top of the other. The documents don't float on top of each other; rather, they fill the available document area as row upon row of thin, horizontal windows. With more than a few documents open, this option displays so little of each page that it's difficult to work on any one page.

- **Tile Vertically (Windows Only).** Just like the previous command, except that documents are placed vertically like stripes going across the screen.

- **Tile (Mac Only).** This has the same effect as Tile Vertically above.

- **Combine As Tabs (Mac Only).** Returns documents that either tile or cascade (see those options above) on the screen into the single, unified tab interface.

- **Next document, Previous document (Mac Only).** This pair of commands let you step through all your open documents, bringing each document in turn to the front of the screen for editing.

- **List of Currently Open Documents.** All the documents that are currently open are listed at the bottom of this menu. Selecting a document brings it to the front for editing. But with the easy document tabs, why bother?

Help Menu

The Help menu offers useful links and references for more information about using, troubleshooting, and extending Dreamweaver:

- **Dreamweaver Help.** The electronic help system that includes a handful of tutorials, background information on Web publishing, and documentation on Dreamweaver's many features.

- **Help Resources Online.** Takes you to a Web page at Adobe.com, where you can download PDF versions of help manuals for Dreamweaver, the Spry Framework, and extending Dreamweaver. There are also links to the online version of the help system, called LiveDocs, which has the added benefit of hosting reader comments—if the help system doesn't have the answer you're looking for, perhaps someone has provided it on this Web site. (Hey, but look in this book first!)

- **Extending Dreamweaver.** An electronic help system for those interested in writing their own "extensions" to Dreamweaver. Extensions are discussed in Chapter 21.

- **Dreamweaver API Reference.** Even more in-depth information for the Extension developer. uber-geeky information on how to communicate directly with Dreamweaver. Programmers only; all others continue, move along, there's nothing to see here.

- **What's New In Dreamweaver.** An overview of the new features introduced in Dreamweaver CS3.

- **Spry Framework.** An online reference to working with and programming Spry widgets like those discussed in Chapter 12. It doesn't have any information on how to use the Spry tools built into Dreamweaver. Instead, it provides more in-depth information for programming-oriented Web designers who wish to jump into Code view, and expand on Dreamweaver's Spry features.

- **ColdFusion Help.** Takes you to an online reference to Adobe's server-side programming language, ColdFusion, on Adobe.com.

- **Reference.** Opens the Reference panel, a searchable guide to HTML tags, Cascading Style Sheets, and JavaScript, as described on page 394.

- **Dreamweaver Exchange.** Launches a Web browser and loads the home page for the Dreamweaver Exchange at Adobe.com. Here you can download extensions for adding new features to Dreamweaver (see Chapter 21 for details).

- **Manage Extensions.** Same as the Manage Extensions menu option listed under the Commands menu (see page 968).

- **Dreamweaver Support Center.** Opens Adobe's online Dreamweaver Support Center Web page in your browser. This Web site offers technical support for known bugs or common questions, downloadable updates to the program, and a link to online forums.

- **Dreamweaver Developer Center.** Opens Adobe's Dev Center—a Web site with tips, tricks, and in-depth articles aimed at Dreamweaver users of all levels.

- **Adobe Online Forums.** Opens an index of available online forums from Adobe's Web site (in your Web browser). You can use the forums to interact with other Adobe customers, post questions, share techniques, or answer questions posted by others. Requires Internet access and a newsgroup reader.

- **Adobe Training.** Opens Adobe's Training Web page, where you can spend even more money learning how to use the program. Cool! Advertising, built right into Dreamweaver.

- **Registration.** Opens a registration form window, so you can register your copy of the program with Adobe. Adobe provides a few free gifts if you register.

- **Activate.** As part of Adobe's attempt to stop piracy of their software, Software Activation contacts Adobe and makes sure that the copy you're using isn't activated on any other computers. You're limited to installing the software on one desktop and one laptop of the same operating system. If you don't activate your software, it won't run on your computer after 30 days.

- **Deactivate.** If you're moving a new computer, *do not forget* to deactivate the software on your old computer. Use this menu option to do it. Deactivating the software lets you install it on another computer.
- **About Dreamweaver (Windows Only).** Opens an About Dreamweaver window, showing your software's version number. (On the Macintosh, this command is in the Dreamweaver menu.)

Index

Colophon

Sanders Kleinfeld and Adam Witwer provided quality control for *Dreamweaver CS3: The Missing Manual*. Ron Strauss wrote the index.

The cover of this book is based on a series design originally created by David Freedman and modified by Mike Kohnke, Karen Montgomery, and Fitch (*www.fitch.com*). Back cover design, dog illustration, and color selection by Fitch.

David Futato designed the interior layout, based on a series design by Phil Simpson. This book was converted by Abby Fox to FrameMaker 5.5.6. The text font is Adobe Minion; the heading font is Adobe Formata Condensed; and the code font is LucasFont's TheSans Mono Condensed. The illustrations that appear in the book were produced by Robert Romano and Jessamyn Read using Macromedia FreeHand MX and Adobe Photoshop CS.

Web Design/Dreamweaver

Answers found here!

Dreamweaver CS3: The Missing Manual. Adobe's powerful Web design program lets you do more than ever. Whether you're sprucing up an old site or building a new one, Dreamweaver has all the tools you need—except a printed guide. Why rely on help menus or books that raise more questions than they answer? This Missing Manual is packed with easy to follow and entertaining guidance.

The important stuff you need to know

- **Getting started**—from building your first site to putting it on the Web.
- **Gain pinpoint control** over design and layout with powerful CSS tools.
- **Add interactivity with forms**, new "Spry" tools, JavaScript, and animations.
- **Build database-driven Web sites** without having to hire a team of geeks.
- **More than 140 pages of step-by-step tutorials** lead you from start to finish.
- **Tricks of the trade** from a veteran Web designer and bestselling author.
- **Design guidance** on what looks good—and what to avoid at all costs.

Why I started the Missing Manual series.

People learn best when information is engaging, clearly written, and funny. Unfortunately, most computer books read like dry catalogs. That's why I created the Missing Manuals. They're entertaining, unafraid to state when a feature is useless or doesn't work right, and—oh, by the way—written by actual *writers*. And on every page, we answer the simple question: "What's this feature *for?*"

David Pogue is a *New York Times* technology columnist, bestselling author, and creator of the Missing Manual series.

US $44.99 CAN $58.99
ISBN-10: 0-596-51043-8
ISBN-13: 978-0-596-51043-5

54499

9 780596 510435

Free online edition
with purchase of this book.
Details on last page.

POGUE PRESS™
O'REILLY®
www.missingmanuals.com