“Exploration is really the essence of the human spirit . . .”

—FRANK BORMAN
EXPLORING DREAM TEMPLATES

Remember the days when you could have only editable and non-editable regions in your templates or when you tried adding a behavior or CSS to a template-based page and it failed? Headaches like these are all but gone with the release of Dreamweaver MX! Many new features have been implemented for templates. I recommend you go get your favorite non-alcoholic beverage, kiss your spouse and children goodnight, call into your boss’s voicemail with your excuse for being late for work tomorrow, and strap into your computer chair because this project will get your creative juices flowing for sure.
IT WORKS LIKE THIS

Dreamweaver MX templates are worth much more exploration and this project builds on Project 2, “Overcoming Template Anxiety,” where you learned how to make a basic template. At the end of this project, you will have a highly customized and functional template set that can be used for site consistency, but also allows you the flexibility to have regions on some pages and not on others.

In this project, you learn many things, including how to add repeating regions (such as repeating tables and repeating rows and optional regions), inserting a MultipleIf expression, making a tag attribute editable, adding CSS to a template-based page, adding a behavior to a template-based page, checking template syntax, and working with methods of detaching template-based pages from a template.

PREPARING TO WORK

Preparing for this project requires the same routine you have repeated with other projects. Copy the Projects03_dream_templates folder on the accompanying CD to your hard drive, start Dreamweaver MX, and define a site using the folder you copied from the CD as the site root.
1 Start Dreamweaver MX and create a site using 03_dream_templates as the root folder. See the figure for a representation of the site settings.

2 Configure Dreamweaver MX to use Dreamweaver MX Workspace (choose Edit>Preferences>General>Change Workspace, select the Dreamweaver MX Workspace radio button, and click OK twice).

Note: I recommend that you use Document Split view to see the code that is being generated as you follow along in this project. (You will need to access Code and Design view in the project as you follow along.) This can be changed by selecting View>Code and Design.

You may notice a lock icon beside the files in the Site panel; this means that the files are read-only. Before you can use these files, you need to remove the lock and make them editable. In the Site window, select the files and folders, right-click, and select Turn Off Read Only. Your local files are now editable!

For this project, I also have included an extension for you; it inserts the required Template comment to turn on/off code above or below HTML functionality as a Snippet. Updates can be found at http://www.prettylady2.net and support email is x10@prettylady2.net. I do plan on adding more template functionality to the Snippets as time permits.

3 Start Extension Manager (EM). Choose File>Install Extension.

4 Browse to the CD>Projects/03_dream_templates/Extensions folder. Select dwm_03_Templates.mxp.
Agree to the license agreement and the extension will install.

**Note:** This extension creates a new folder called **Templates** in your Snippets panel and adds four Snippets. The Snippets include these:

- Ability to add a Template comment that enables/disables code above/below HTML (as specified in the help docs).
- Required base code for a **MultipleIf** conditional region, also as specified in the help docs.

We will be referencing some of these Snippets later in the project, so be sure to install this extension.

### Repeating Regions—Rows

A repeating region can be used to allow the builder of the template-based page to add table rows or other types of repeating regions without having to edit the template. This allows for the template builder to have more control of the template while allowing the user of the template-based file the freedom to add content without messing up the structure of the template. In this project segment, you learn how to add, remove, and use a table row repeating region in your template and template-based page(s).

**Tip:** When naming regions, each region name must be unique.

1. Open the file named **Rep_Row_Template.dwt** in the **Templates** folder.
2 Position your cursor in the table cell that says Company Info 2. Using the Tag Selector, choose the right-most \(<tr>\) tag.

3 From the menu, choose Insert>Template Objects> Repeating Region.

4 Type MenuRow1 in the New Repeating Region dialog box and click OK.

5 Select the content of the cell and from the menu, choose Insert>Template Objects>Editable Region.

6 In the Editable Region dialog box, type the name MR1Content and click OK.

7 Repeat Steps 1 through 6, selecting the Section 4 cell, using MenuRow2 as the Repeating Region name and MR2Content for the Editable Region name. Save the template.

8 You should be prompted to Update Template Files that are based on this template. Choose Update. Close the update report dialog and continue on to Step 9.

9 Close the template file.

10 Open the root file named Repeat_Row.htm. This file is based on the template you worked with in Steps 1 through 9, Rep_Row_Template.dwt.

11 Notice that there is a new GUI where you added editable rows. Position the cursor in the MR1Content cell that is named, select the cell contents, and type Additional1 (or use your own text).
Click the GUI + button to add another row; then select the content of this row and replace it with Additional2 or use your own content.

Repeat Steps 11 and 12 for the MR2Content repeating row using Additional3 and Additional4 as your entered text.

You may have to select the content and set it as a paragraph for proper cross-browser rendering. To do this, open the Property inspector, click the Format drop-down menu and choose Paragraph.

Save the document, preview in a browser to see that your edits are viewable, and then close the document. You have successfully added a repeating region, made it editable, and controlled it from the template-based page.

So now you decide that you don’t want the repeating editable regions anymore, and now they have to be deleted. Open the template Rep_Row_Template.dwt.
16 To remove an inserted editable region, select the editable region you want to remove by clicking inside the editable region. Then select the `<mmtemplate:editable>` tag using the Tag Selector. Use the Modify>Templates>Remove Template Markup menu item to remove the selected editable region. At this point, remove the MR1 Content editable region.

17 To remove an inserted repeating region, select the repeating region you want to remove by clicking inside the repeating region, and then select the `<mmtemplate:repeat>` tag using the Tag Selector. Use the same menu item (Remove Template Markup) to delete the repeating region. At this point, remove the MenuRow 1 repeat region.

18 Repeat Steps 16 and 17 to remove the other editable and repeating regions you added to the page.

19 Save the document, update the pages associated to this template, and close the template. You will be prompted to relocate found content in the Inconsistent Region Names dialog. Select the MenuRow 1 and choose nowhere from the drop-down list. Repeat this for MenuRow 2. This action makes sure that they are completely deleted from the template-based page. Close the template file.

20 Open Repeat_Row.htm again to verify that the repeating row and editable region are removed from the page. Close the file when verified.

With this segment, you have learned how to add a repeating row, make it editable, and modify a template-based page to use the repeating editable region. You then decided that you wanted a change and deleted the editable region and the repeating region.
REPEATING TABLES

Although this segment is similar to the previous one, there are important differences. A repeating table has a complete row that is marked as repeating and each cell in that row is an editable region. Another difference is that instead of using the `<tr>` tag as your selector, you use a friendly user interface (UI) to perform the insertion of the Repeating Table object.

1. Open the file named `Rep_Table_Template.dwt` in the Templates folder.

2. Position the cursor in the large center cell with a white background (to the right of the menu system).

3. From the menu, choose Insert>Template Objects>Repeating Table. Fill in the following properties in the Repeating Table dialog box and then click OK:
   
   - Rows: 2
   - Cell Padding: 5
   - Columns: 3
   - Cell Spacing: 0
   - Width: 75 Percent
   - Border: 1
   - Starting Row: 2
   - Ending Row: 2
   - Region Name: DataTable1

Tip: Region names cannot contain spaces or any special characters; you can use only alpha-numeric characters.
4 In Code view, change the top row \(<td>\) tag pairs to \(<th>\) to make them table headers instead of standard table cells.

This includes code lines 122 through 124, where you change the cells from \(<td>&nbsp;</td>\) to \(<th>&nbsp;</th>\). You do this to make the row your table heading row. You also may decide you want column 1 to be the headings. Changing from \(<td>\) to \(<th>\) centers the content and makes it bold by default.

5 Now back to Design view. In row 1 (which is now table headers), type the following in the cells, beginning on the left and moving to the right: **Item No.**, **Item Name**, and **Item Description**.

To do this, position the cursor in the appropriate cell and type the text; then move the cursor to the next desired cell and repeat.

6 Click the region header labeled EditRegion3 in the Design window (this is your editable region label), open the Property inspector panel (the name should say Editable Region), and change the contents of the Name field to **ItemNo**.
**Note:** You can display the template region Property inspector three different ways when you are editing a template (not a template-based file):
- Using the Tag Selector, select `<m:template:regontype>`.
- Select the `<!-- TemplateEndRepeat -->` comment in Code view. (This is the closing comment for the region. Note that this method does not work for most region opening comments!)
- Carefully select the region header in Design view. (Note that you have to be very careful in selecting when you use this method because you could easily select the parent region by mistake.)

I recommend using the first method because it is the most reliable.

7 Repeat to the right, naming the remaining two fields **ItemName** and **ItemDesc**.

8 Save the template. (If you are prompted to update the template files that are based on this template, choose Update. Close the Update Report dialog box.) Close the template.

9 Open the root file named **Repeat_Table.htm**. Notice that the row contains an editable region in each cell and that it looks identical to the repeating region row discussed in the previous segment.

Position the cursor in row 2, column 1 (this should be the first editable region named **ItemNo**) and type **03**, then press the Tab key and type **Ring** in the editable region named **ItemName**. Press Tab again and type 1 karat **Diamond Cluster** in the editable region named **ItemDesc**. Save the page and preview it in a browser.
10 Click the + button in the repeating table GUI three times to add three more rows to the table. Starting in the third row, insert the data displayed in the list to the right:

<table>
<thead>
<tr>
<th>Item No</th>
<th>Item Name</th>
<th>Item Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Watch</td>
<td>Analog</td>
</tr>
<tr>
<td>04</td>
<td>Bracelet</td>
<td>Gold</td>
</tr>
<tr>
<td>02</td>
<td>Necklace</td>
<td>Silver</td>
</tr>
</tbody>
</table>

Note: You can use the Tab key to move from field to field and then to the next row just like a normal table. You cannot, however, add another repeating table row by using the Tab key because an error message will be displayed.

When you’ve finished entering the data, save the page and preview it in a browser. Your item list should be 03, 01, 04, and 02 as you look from top to bottom in the Item No. column. Notice that the items are out of order! We’ll fix that in the next few steps. Close your browser session and go back to Dreamweaver.

11 Position your cursor in the Item No. 03 row and click the down-arrow of the Repeating Object GUI once. This repositions the currently selected row down one position each time it is clicked. The order should now be 01, 03, 04, and 02.

12 Now position your cursor in the Item No. 02 row and click the up-arrow of the Repeating Object GUI twice. This moves this row up two positions. The order should now be 01, 02, 03, and 04. Save the page and close it.

You’re probably wondering what the difference is between these last two objectives. With a repeating table, you can define more than one row and use an interface to control which row is the repeating row with an interface; it also makes every cell in the defined row editable. A repeating region, on the other hand, only has one row, but the interface can be set so that every cell in that row is editable.

Tip: If you know your Optional Region is going to be editable as well, use Insert>Template Objects>Editable Optional Region.
other hand, is selectable, which means that if you were to select the entire row, you could make only the last two cells editable and leave the leftmost cell non-editable so that cell remains the same down the table. (This is perfect for working with bullets, for example.) Repeating regions don’t have to be a row; they can be an image or just about any tag. (Lists provide a good example for a repeating region.)

**Inserting Optional Regions**

In this section, you learn how to insert an optional region in your template, set it as viewable or hidden, and control it in your template-based page.

1. Open the file named *Opt_Region_Template.dwt* in the *Templates* folder.

2. Select the abstract art image in the large center cell with a white background. Select *Insert* > *Template Objects* > *Optional Region*.

3. In the Optional Region GUI, give the region a name of *ArtPicture1* and place a checkmark in the Show by Default field if it is not already checked. (The Advanced tab is not used at this time.) Click the OK button.

4. In Code view, scroll to just above the closing head tag </head> and you should see the following code added to your document (line 38 in the top figure).

   If you select the image and view the code, you will see the code shown in lines 124–126 of the figure on the bottom. (I have modified the document for the purpose of the screenshot, so your view may be different than the code around line 122 in Code view.)
Save the template. (If you are prompted to update the template files that are based on this template, choose Update. Then close the Update Report dialog box.) Close the template.

6 Open the root document Opt.Region.htm. Notice that the editable regions are there. Select Modify>Template Properties.

7 In the UI, select ArtPicture1. By unchecking the Show ArtPicture1 checkbox, the image will not be shown. Notice that the source template provides only the initial state of the image, and the htm page itself controls the other state of the image—in this case, off. Remove the checkmark from the Show ArtPicture1 checkbox, save your document, and preview it in a browser.

8 Repeat Step 7, this time placing a checkmark in the Show ArtPicture1 checkbox; then save the document and preview in a browser. When you are satisfied with the results, close the document in Dreamweaver.

You have just successfully created and used an optional region. Congratulations!
INSERTING A *MultipleIf* CONDITIONAL STATEMENT

So, you want to have a specific image shown on certain pages—or not at all! To do this, you use an optional region and convert it to a *MultipleIf* conditional statement.

1. Open the file named `Multiple_If_Template.dwt` in the `Templates` folder.

2. In Code view, insert the highlighted line of code shown just above the closing head tag. You can use the Snippet in the Snippet Template folder called `MultipleIf Expression` and then modify it to match the highlighted code on the right, or you can open `MI_Head.txt` and copy and paste it into the head region of the template file.

3. Select the abstract art image in Design view and go to Code view. Notice that the code for the image is highlighted. Press the Delete key to delete the image reference in the template file.

4. Press the Ctrl+` keyboard shortcut (Cmd+` for Mac) to go to the Code window of the Split view. (Your cursor should be positioned beside the `<td>` tag, where the image was located.) Type the highlighted code listed at the right. Alternatively, if you don’t want to type all the code, you can open `MI_Body1.txt` and use the copy and paste operations to insert the code into the template file.

The code you just entered says this: “Provide the end user of the template-based file with the choice of two images. If image 1 is set, display image 1 and hide image 2. If image 2 is set, display image 2 and hide image 1.”

**Tip:** *MultipleIf* conditional regions must be entered by hand, but you can use the Advanced tab of the Optional Region dialog box to generate your conditions. If you installed the extension for this project, your Snippets>Templates folder will display the head and body regions as usable Snippets.
What you’ve done here is manually enter a conditional region. In this instance, all the images are defined in the template; the end page allows you to choose which image to show. This is an excellent way to define a specific image for, say, a support department and a sales department not yet using the exact same layout.

**Note:** Notice that both images are shown in the template. This is the best presentation because you are editing the template file and you need to see the options available while editing. This won’t be the case in the template-based page, though, as you will see below.

5 Save the template. (Choose Update if you are prompted to update the template files that are based on this template; then close the dialog box.) Close the template.

6 Open the root document *Multiple_If.htm*. Notice that the editable regions are there, but wait and select Modify>Template Properties.

7 In the UI, select ArtPicture1, change the value from 1 to 2.

Because you have a choice of two images, placing a value of 3 or higher there will work, but no image will be displayed when you view the page online or in a local browser! Try it, insert any number other than 1 or 2 and see what happens!
You can correct this by opening the template file (Multiple_if_template.dwt) and manually adding another conditional test using the following line (outlined to the right) just before the closing <!-- TemplateEndMultipleIf --> statement. Add the highlighted code at right. If you don’t want to type all the code, open the file named MI_Body2.txt and use copy and paste to place the code into your template file.

What this does is tell Dreamweaver to display a text message in the template-based file if neither option 1 nor option 2 is selected. If you used more images or objects, reflect this in the conditional test. What you have done here is add a conditional statement that checks the template-based page template property of ArtImage. If this property is not set to a 1 or a 2, the inserted message is displayed. If the value is a 1 or a 2, the appropriate image appears.
9  Save the template (choose Update if prompted). Close the Update Report dialog box. Close the template.

10  Open the root document Multiple_If.htm and modify the Template Properties, trying various values from 1 through 5, saving the page and previewing in a browser each time.

11  Click OK, save the document, and preview the page in a browser.

12  Close any open documents and continue to the next segment.

Here, you can see that the template allows you to define template parameters to, in this instance, control which image shows on which page. The template allows you only to define the choice of images and the page itself. End users can use Template Properties to define the desired image without messing up the layout by editing the template itself.

**EDITABLE TAG ATTRIBUTES**

You want to allow the page-building personnel to have some creative input into the design without risking corruption of the template? Great, let’s add some editable tag attributes to the mix to get control over the `<img>` tag’s `alt` attribute content.

Note: Always use Modify>Template Properties to modify the desired MultipleIf element; otherwise, elements could be selected or displayed incorrectly.
1. Open the file named Edit_Attrib_Template.dwt in the Templates folder.

2. Select the abstract art image and choose Modify>Templates>Make Attribute Editable. The Editable Tag Attributes dialog box opens.

   The Attribute field populates with a drop-down of found attributes for the selected tag. If an attribute is not available in the list, and you know it is specific for the tag, click Add and type the attribute name.

3. Click the Add button and in the dialog, type ALT and click OK. Notice that the Make Attribute Editable checkbox was automatically checked by this action!

4. In the Label field, type Alt1. In the Type dropdown, choose text, and in the Default field, type Alternate Text and then click OK.

5. In Code view, you see that the image tag now has an alt attribute and it is defined with @@(Alt1)@@. If you also look in the head section just before the </head> tag, you will see that the following code has been added:

   ```html
   <!-- TemplateParam name="Alt1" type="text"
   value="alternate text" -->
   ```

   Oh look, another parameter type!

6. Save the template (choose Update if you’re prompted to do so; then close the dialog box). Close the template.
7 Open the root document Edit_Attrib.htm. Notice that the editable regions are there and select Modify>Template Properties.

8 In the UI, select Alt1, and change the value to Abstract Art. Click OK, save the document, and preview in a browser.

9 Repeat Steps 2 through 8, this time selecting other attributes of the image tag to make them editable. Define their defaults and experiment! Save the document and preview it in a browser. Close the document when you are satisfied with the results.

This is a great way to add a customizable link in your document to allow a template page author to insert links. Experiment with this; try making it a repeating region and see what you can do with it.

**Adding CSS to a Template-Based Page**

In the days of Dreamweaver 4, you could not add or edit a style block of a page that was based on a template (without hand-coding a workaround, that is). Well, that feature is built into Dreamweaver MX now, and you will learn and use that method here.

1 Open the file named Edit_Attrib.htm in the root folder.

---

**Note:** You cannot add CSS to a non-editable region of a page. To add CSS to a non-editable region, it must be done in the template itself.

**Tip:** You can add an editable tag attribute in the template to make the desired tag accept user-added CSS!
2. Open the CSS Styles panel, and click the Edit Styles radio button. Choose the New CSS Style button from the bottom of the panel and choose the Make Custom Style (Class) radio button. Type .testpara in the Name field, and lastly, in the Define In Option, click the This Document Only radio button and click OK.

3. Under Category, choose Type, and choose the following settings:
   - **Font:** Times New Roman, Times, serif (Your font choices may vary depending on the platform, OS, and fonts installed on your machine.)
   - **Size:** 14 Pixels
   - **Style:** Italic
   - **Color:** #008080
   Click Apply and OK to close the CSS Editor.

4. Now view your document in Code view and look at the page, specifically, at an editable region named head in the head block of the document, where you will see the class added inside a new style block. Dreamweaver MX automatically adds this editable region to templates and template-based documents for just this reason as well as adding behaviors.

```html
<instanceBeginEditable name="head" -->
<style type="text/css">
instanceBeginEditable -->
```
5. Using Design view, position your cursor in the first paragraph of the Body_Content editable region and select the `<p.Contenttext>` tag using the Tag Selector.

6. Select Text>CSS Styles>testpara. and your document view will be refreshed with the selected style.

7. Save the document, and preview it in a browser. Close the file when you are satisfied with your changes.

### ADDING A BEHAVIOR TO A TEMPLATE-BASED PAGE

With the advent of editable tag attributes, you can add behaviors to the body tag with ease, as long as the JavaScript already exists on the page and you know what the function name and parameters are. (It pays here to apply your behavior to a non-template-based page and use copy and paste to insert the required code in the Template Properties dialog box.)

1. Open the file named Behavior_Template.dwt in the Templates folder.

**Note:** A few notes about behaviors:
- You can add behaviors to any body-area editable region without issue.
- If you want to add a behavior to a non-editable region, the appropriate tag must have an editable tag attribute. (See the section, “Editable Tag Attributes.”)
- If the behavior JavaScript block does not exist on the page when you add the behavior, the JavaScript will go in the same editable region (called head) as the page added CSS as in the previous segment.
2 Select the body tag and make it editable as you have done previously. For the editable attribute, choose onLoad. (If you don’t see onLoad in the drop-down list, click the Add button, type onLoad, and click OK.) Label the tag LoadIt and leave the remainder of the dialog fields at their defaults; then click OK.

This does not assign a value to the onLoad event, but rather assigns a parameter that you can use from the Template-based page to the onLoad event.

Save the template, updating the files if prompted. Close the template.

3 Open the file in the root folder named Behavior.htm.

4 Select Modify>Template Properties, choose LoadIt from the dialog box, and in the text field, type MM_showHideLayers(‘LyrCopyr’,”, ‘show’). Click OK.

5 Save the document, preview it in a browser to see the copyright message layer shown onLoad at the top of the page. Close the document when you are satisfied with the changes.

You can select just about any element and apply a behavior to it, but it has to be in an editable region or an editable attribute in the desired element.

---

**CHECKING TEMPLATE SYNTAX**

You now have the capability of manually checking your template syntax prior to saving the template or Dreamweaver will do it for you automatically when you save a template. If any errors are detected in your template syntax, they are reported to you in a dialog box showing the error description, line number, and the location of error.

1 To check your template syntax manually, choose Modify>Templates> Check Template Syntax.

2 To automatically check your syntax, save the template file.
3 If any errors are detected, the syntax checker dialog box appears. When you are manually using the Check Template Syntax functions, if everything is OK, the OK dialog appears. Note that when you “automatically” save your template, you will not see the OK dialog box if everything is fine.

**Detaching Your Template-Based Pages**

Remember in Dreamweaver 4 when you had to detach the page from the template to add a custom behavior to the body tag? That functionality is still there, but you have a new way to do it without completely messing up your template-based page. This segment of the project deals with methods of detaching your pages from a template.

You really have no reason to detach your pages from a template anymore due to functionality reasons (such as adding a behavior to the body tag); however, you might have other reason to detach your pages. These reasons include, but are not limited to the following:

- To reduce the amount of code in your page, thereby reducing bandwidth usage.
- To make troubleshooting your page(s) easier for outsiders who might not have your template.
- To camouflage the fact that you used a template to build the site.
- To make a unique change to one instance of the template, that you do not want applied in other instances in the same template.
Method A: Clean Up HTML

1. Open the file name methoda.htm in the root site folder.

2. From the menu, choose Commands>Clean Up HTML. The Clean Up HTML/XHTML dialog box appears.

3. Place a checkmark in Remove>Dreamweaver Special Markup. When you do this, a warning message appears telling you that library items, templates, and tracing images will no longer be updated. Click OK.

4. The Clean up summary report lets you know how many tags were affected on the page. Click OK.

5. Dreamweaver refreshes the page in your original view. The page is 100 percent editable—no more locked regions.

6. Save the resulting page as methoda_done.htm and close the document.

When to use this method: Use this method when you are working on a single page that is based on a template, if you want a level of control to delete all template markup and do a code check at the same time.

Method B: Detach from Template

1. Open the file name methodb.htm in the root site folder.

2. From the menu, choose Modify>Templates>Detach from Template.

3. Save the resulting page as methodb_done.htm and close the document.
When to use this method: Use this method most for single-page detachment. The reason for this is that there are no dialog boxes to complete and no warning prompts—it’s instantaneous.

Method C: Export Without Markup

1 Using your file manager application, create a new folder on your local hard drive and label it methodc. (Do not create this folder in the site structure or errors will occur!) Close your file management application and start Dreamweaver if necessary.

2 You can open any document or not open any at all for this procedure. It’s your choice—either way, the result is the same. For the purpose of this demonstration, open methodc.htm from the site root.

3 From the menu, choose Modify>Templates>Export without Markup.

4 The Export Site Without Template Markup dialog box appears.

5 Using the Browse button, navigate to the methodc folder, open it, and click the Select button in the Extract Template XML dialog box.

6 Optional: Enable Keep Template Data Files by placing a check in the checkbox, if it is empty. This exports the template data to an .xml file, which makes it easier in the future to apply the removed template to the page again. It also helps Dreamweaver decide if the page needs updating when the Extract Only Changed Files is enabled. If you decide to disable this, no .xml files will be generated and it will be difficult to recreate or apply the template to the pages again, and Dreamweaver won’t know the change status as discussed in the following steps.
Optional: Enable Extract Only Changed Files by placing a check in the checkbox, if it is empty. (This exports the files that have changed since your last export or the entire site if no export has been previously performed.)

Click OK. Dreamweaver’s processing dialog may show up with the cursor in the busy state and after a little bit the dialog goes away. At this point, your entire site is exported with no visual clue even if you had a document open at the time.

Close the document if you had one open.

Using your file management application, navigate to the folder methodcand have a look. Now there is an additional.xml file for each template-based page. These.xml files are used by Dreamweaver and Dreamweaver alone and do not require publishing to the web site on the server. You could delete them if you like, but the reasons for keeping them are detailed above if you decided to create them in the first place.

Close your file management application.

When to use this method: Use this method when you want to be able to delete all template markup on the entire site or only on changed files since last export, and you don’t want to affect your local site files.

Cautionary Note: Unless you are careful with the options, you could export newer files that you don’t want updated, so choose your options carefully. All template-based pages will have their template code removed using this method, excluding the templates themselves.
Here you have learned various ways to detach a file or a group of files from the template with which they were created. Exercise caution using any of the methods; it is easy to overwrite good pages with bad and lose the template update capability in a site.

Play with the various methods you have learned and decide for yourself when you should use which method for your own needs and situation.

MODIFICATIONS

You can go in many different directions with this template set, which contains two basic templates. One template could be used for the index page, a news area start page, or just about any sectional starting page. The other is useful for the remainder of the site. For instance, you could make a nice PC service web site out of this template or perhaps a web design site. All you need to do is modify the provided template to your requirements, but try adding CSS styles to the repeating and optional regions to further customize it. Any element of the template set can be customized by combining what you have learned throughout this project into one template that controls your whole site.

Not only can you modify the templates, but you now have the knowledge from Projects 2 and 3 to create a completely new template that matches your site design requirement. But wait, there’s more... check out the help files (press F1 when Dreamweaver is active) and do a search for the different regions of a template because you can do much more with templates than you’ve seen here. To give you a teaser, think about having the capability of programmatically telling a table row to alternate colors. And we didn’t even get a chance to touch on the completely new Nested Templates.

I’m sure that after you read Projects 2 and 3, the help files that you will be making your own templates with will be far superior region conditional statements than the ones we have explored here. Don’t be afraid to experiment!

Enjoy...

Tip: If you have a dynamic site that was created with a server-side language such as ASP, PHP, and so on, you are required to use the following code Snippet in the Template file:

```
<!-- TemplateInfo codeOutsideHTMLIsLocked="true" -->
```

This code must be typed, or you can use the extension to add the Snippet to your Snippet panel. The reason for this is that server-side technologies often add code above the opening `<html>` tag, and adding this line ensures this server-side code will be transferred to the template-based page(s).

If your template-based page contains no server-side code, you don’t need to add this line. Also note that you can still add server-side code to template-based files without issue.

Tip: Use the Multiple If Template Snippets to save yourself some typing when using these conditional regions.