Creating a Simple Blog Part 4

By: Tom Muck

The first three parts of this series showed how to create a simple blog using the basic functionality of Dreamweaver in all server models. In addition to building a blog, this series is intended to acquaint you with basic Dreamweaver built-in functionality, such as server behaviors and application objects. One of the great things about using DW server behaviors is that they are easily modified. This part of the series will describe how to add categories to the blog by modifying the pages you've already created and also how to display the number of comments on each item. If you haven't read the first three parts, they are located here:

Part 1 (http://www.communitymx.com/abstract.cfm?cid=7CC52)
Part 2 (http://www.communitymx.com/abstract.cfm?cid=C6355)
Part 3 (http://www.communitymx.com/abstract.cfm?cid=5B658)

Adding Categories

Blog categories allow you to create items under a specific category, which allows your users to view only those categories that interest them. For example, if you have a personal blog with entries about fishing, bowling, television, and recipes, if someone was only interested in bowling, they would be able to click a link and see only entries pertaining to bowling. That person has a sad life indeed, but we'll add the functionality to the blog to make it easier for him.

The functionality consists of two parts -- the administration section to add a new category, and the display section which will allow a link to be clicked to filter the blog entries by that category. Before we begin, we'll modify our database to add the categories table, and modify the **blog_items** table to include categories. The following is for SQL Server:

```
CREATE TABLE blog_categories (
  category_id int IDENTITY (1, 1) NOT NULL ,
  category_name varchar (50) NULL
)

ALTER TABLE blog_items
  ADD blog_category INT NULL
```

The following is for MySQL:

```
CREATE TABLE blog_categories (
  category_id int AUTO_INCREMENT NOT NULL, PRIMARY
KEY(category_id),
  category_name varchar(50) NULL
) ;

ALTER TABLE blog_items
  ADD blog_category INT NULL;
```

You can also create/modify the tables by hand using the SQL Server Enterprise Manager, or MS Access table designer. With the new data structures in place, we can begin on the admin section.

Administering Categories

If you recall, we had an administration section in the site that allowed you to view items, insert items, and update items. We'll create a page now to add categories. Create a page named **categories.php** (or .cfm, .asp, or whatever server model you are working with. I'll be referring to php pages, but you should use your own server model.) On that page, create a recordset named **rsCategories**. The recordset should use this SQL to list out all the existing categories:

```
SELECT * FROM blog_categories ORDER BY category_id
```

Next, add a Dynamic Table (**Insert > Application Objects > Dynamic Data > Dynamic Table**) to the page using the **rsCategories** recordset. Check the **Show All** radio button to display all categories.

Finally, add a Record Insertion Form Wizard (**Insert > Application Objects > Insert record > Record Insertion Form Wizard**). The object should use the following parameters (parameters vary by server model):

- **Data Source**: your CMXBlog database
- **Username**: Your database username
- **Password**: Your database password
- **Table**: blog_categories
- **After inserting, go to**: categories.php
- **Form Fields**: Use the minus button to remove the **category_id** field, and set the following field:

Column	Label	Display As	Submit As
category_desc	Category:	textfield	Text

The resulting page will display all categories, and also allow you to add new categories to the page, as shown in Figure 1:

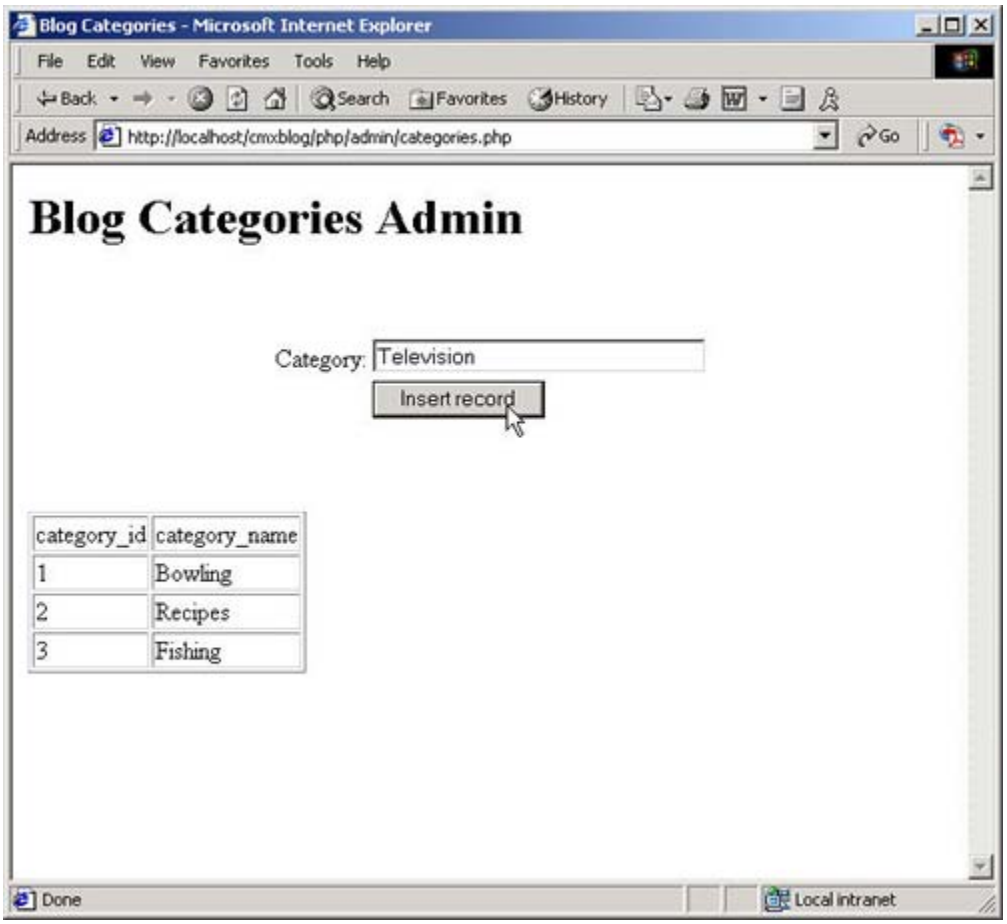

Figure 1: *The blog categories administration page*

After creating the administration page, you'll need to modify the other admin pages so that you can now use categories. First, add the category field to the **insert.php** page. Make the following modifications to the insert page:

1. To make it quick and painless, first open the **categories.php** page one more time. Copy the **rsCategories** recordset from the Bindings panel (Right-click > Copy). Open the **insert.php** page and paste the recordset. This recordset will feed a dropdown list for the insert.
2. Next, add a table row to the insert display so that you can add a dropdown list.
3. Insert a dropdown list into the new, blank table cell, along with some descriptive text. Name the dropdown list **category**.
4. Select the dropdown and open the property inspector. Click the Dynamic button on the inspector and fill in the values shown in Figure 2
5. Finally, double-click the **Insert Record** instance already in the server behaviors panel. The new dropdown list that you added to the page is not inserting anything yet, and the **blog_category** field should show up in the dialog box with "Does Not Get a Value" next to it. Change this item so that it receives a value from the **category** dropdown. It should submit as **Integer**.

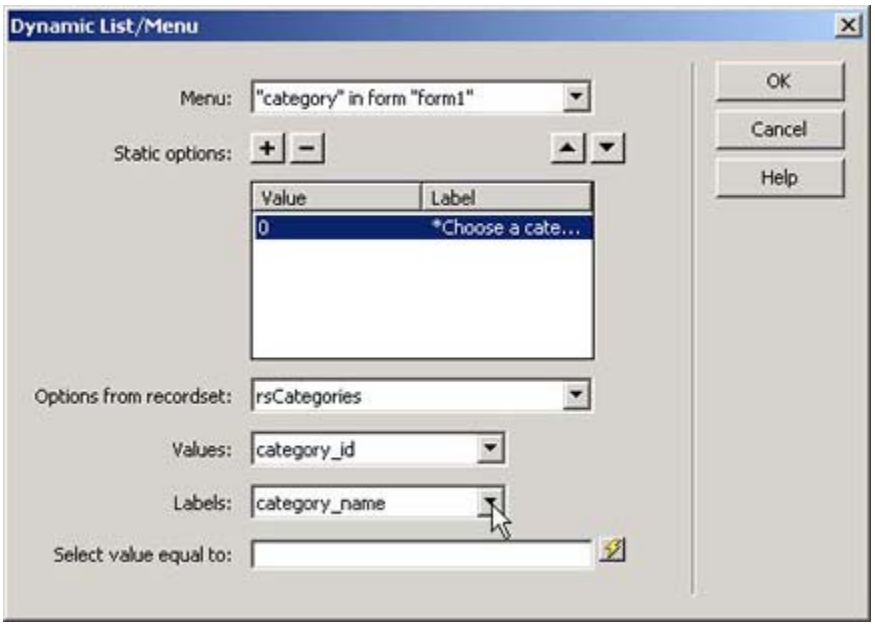

Figure 2: *Values for the dynamic select list*

That is all you need for the insert page. Next, on the index page of the *admin* section (don't modify the index page of the main site yet!), make the following modifications:

1. Add a link to the new **categories.php** page
2. Modify the recordset to include a join to the new **blog_categories** table (using the Advanced tab of the Dreamweaver query builder). This modification will show the related category for the blog item:

```
SELECT blog_item_id, blog_item, blog_item_datetime,
blog_item_title, category_name
FROM blog_items
INNER JOIN blog_categories
ON blog_items.blog_category = blog_categories.category_id
ORDER BY blog_item_datetime DESC
```

3. Drag the **category_name** field onto the blog items display on the page next to some descriptive text.

The display should now look like Figure 3 (assuming you have entered data that has a category):

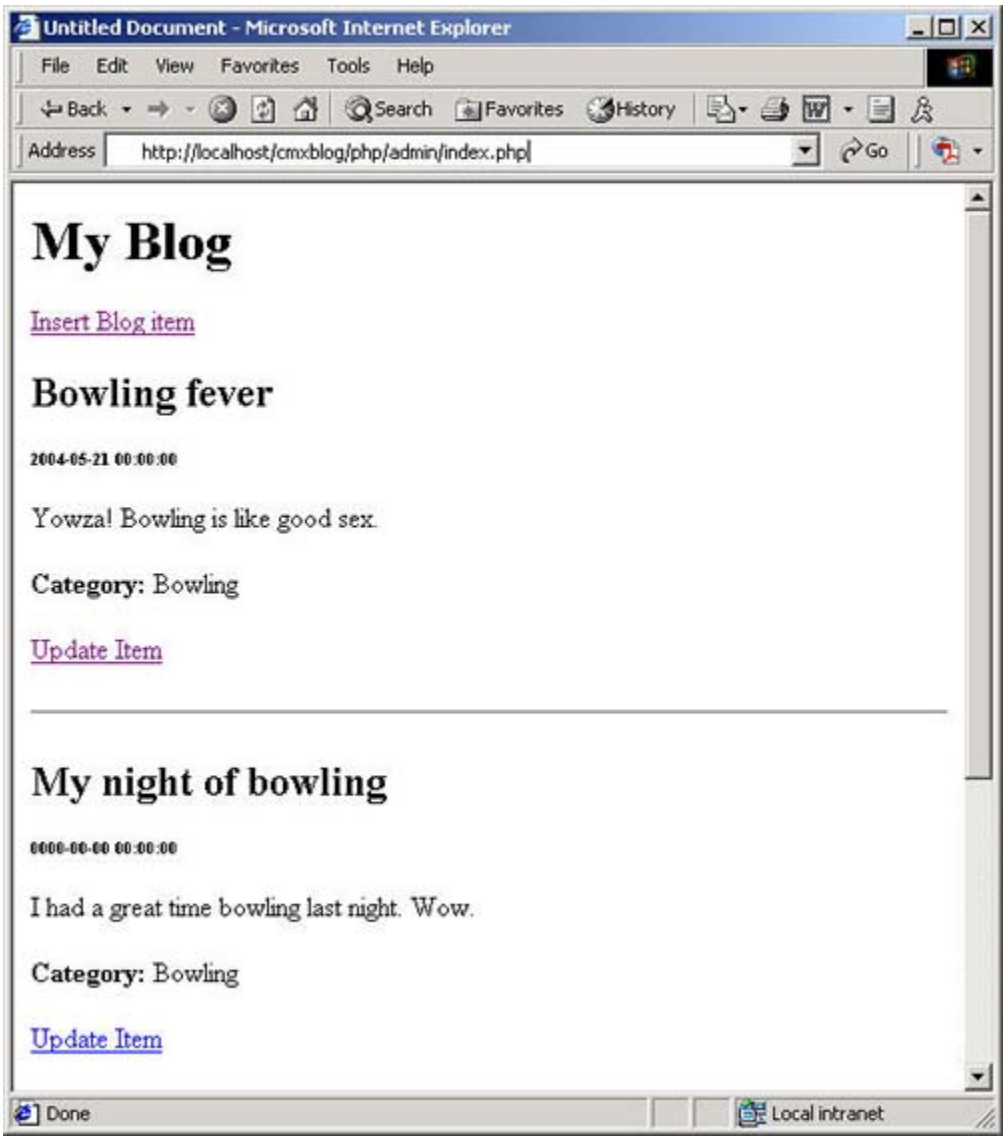

Figure 3: *Updated blog admin now showing categories*

Creating a Category Page for Display

Just as we added an archive page to simplify the display of a single item, we'll create a **category.php** page to simplify the displaying of blog items within a specific category. First, we'll modify the index page to add links to categories. Follow these steps to add a list of categories:

1. First, just as you copied/pasted earlier, go back to the categories page in the Admin section and copy the rsCategories recordset. Paste it into the index page for the main blog site.
2. Find a spot on the page where you want to list the categories and add a heading ("Categories") and drag the **category_name** field from the **rsCategories** recordset below it somewhere.
3. Go to the property inspector and change the **category_name** field to an unordered list
4. Add a Repeat Region to the field using the rsCategories recordset, showing all records. This will set up a dynamic list fed by the recordset.
5. Finally, add a link to **category.php** (which we'll create next.) The link should pass a URL variable named category using the **category_id** field from the **rsCategories** recordset.

The easiest way to do that is from the Property Inspector, by clicking the folder icon. You can then type in the name of the file (**category.php**) and add parameters from **rsCategories**, as shown in Figure 4.

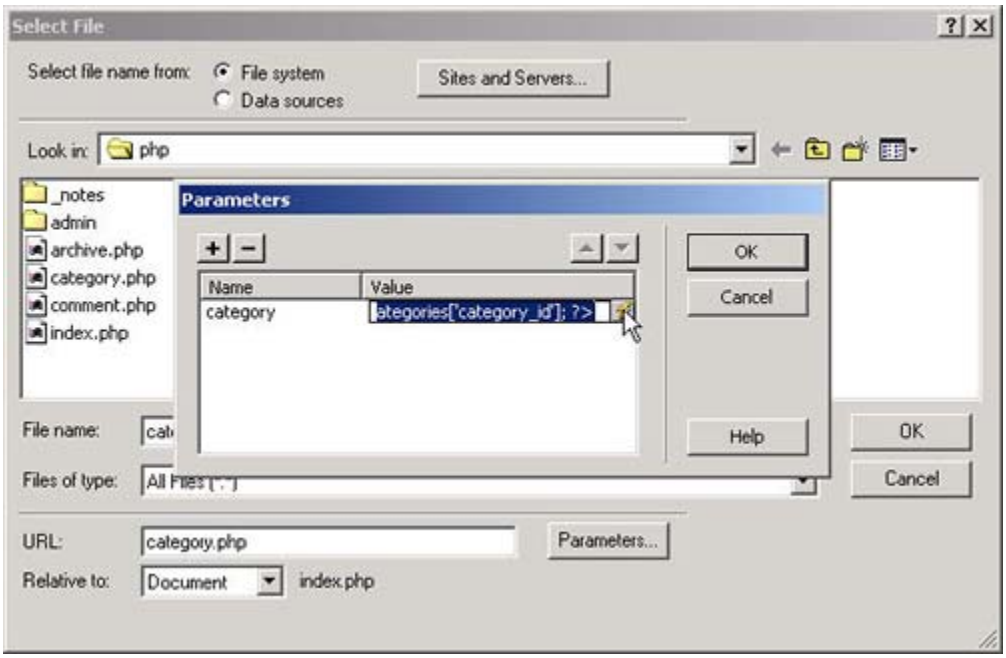

Figure 4: *Adding a link with parameter to the category page*

Next, we'll create the category page. Create a new page at the root of the blog site called **category.php** that is an exact copy of **index.php** (from the main site -- not the admin section) and follow these steps:

1. Open the **rsBlogItems** recordset by double-clicking it. Using the Simple recordset dialog, add a url parameter named Category as a filter for **blog_category**, as shown in Figure 5. Also, make sure the **blog_category** field is selected as well. Click OK.

If you have already modified the main index page to show categories (copying the steps you used to modify the index page in the admin section), you will not be able to use the simple recordset builder. Revert your category.php to match the index page shown in part 1 of this tutorial.

2. Create a new **rsCategory** recordset using the simple recordset dialog. It should show only one field (**category_name**) and should also be filtered by the url parameter named category.
3. Drag the **category_name** field from the new **rsCategory** recordset to the page next to the heading.

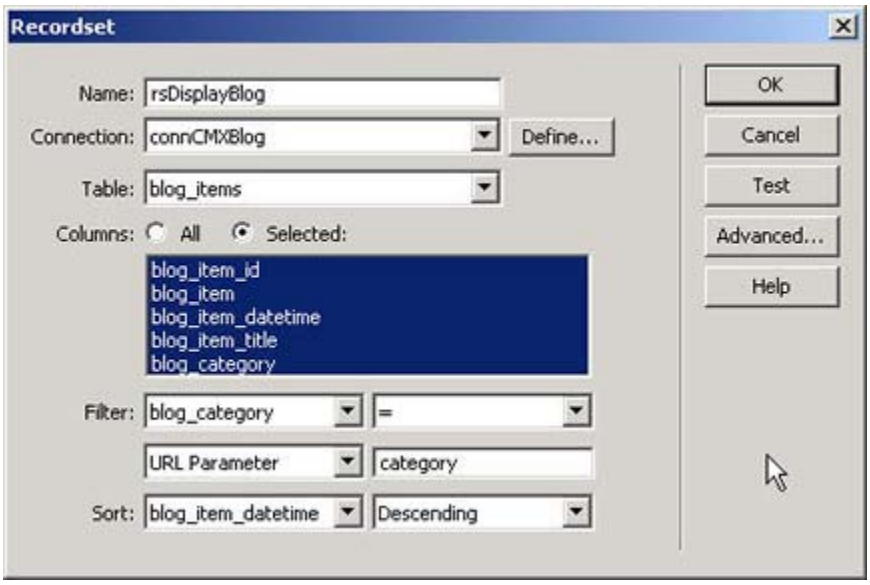

Figure 5: *The simple recordset dialog is used to add a filter to the rsDisplayBlog recordset*

The finished category page should look something like this:

Figure 6: *The finished category page*

The other pages that require changes are update.php (in the admin folder), index.php (in the main site),

archive.php, and rss.php. You can use the same techniques shown here to modify each of these pages.

Adding a Comment Count

On the index page, you have a listing of blog items and a mechanism for adding comments; however, you have no display of how many comments each item has on it. This can be done with a little SQL. Open the index.php page in the main site, and double-click the rsDisplayBlog to modify it. If you already modified it to display the category name, the SQL will look like this:

```
SELECT blog_item_id, blog_item, blog_item_datetime,
blog_item_title, category_name
FROM blog_items
INNER JOIN blog_categories
ON blog_items.blog_category = blog_categories.category_id
ORDER BY blog_item_datetime DESC
```

We'll modify the SQL to add a field alias that displays the count of blog comments received on a particular blog item. To do so, we'll also have to put aliases on all the tables referenced in the SQL as well (changes highlighted):

```
SELECT i.blog_item_id, i.blog_item, i.blog_item_datetime,
i.blog_item_title, bc.category_name
, COUNT(c.comment_id) AS TheCommentCount FROM blog_items i
LEFT OUTER JOIN blog_comments c
ON c.blog_item_id = i.blog_item_id
INNER JOIN blog_categories bc
ON i.blog_category = bc.category_id
GROUP BY i.blog_item_id, i.blog_item_datetime, i.blog_item,
i.blog_item_title,
bc.category_name
ORDER BY i.blog_item_datetime DESC
```

Read the article on SQL formatting (http://www.communitymx.com/content/article.cfm?cid=C138F) at Community MX for an overview on how to work with table and field aliases to make SQL code more readable.

Next, simply drag the new field (TheCommentCount) to the page within your repeat region next to some descriptive text to display the count of how many comments you have on each blog item.

Conclusion

In this installment you learned how to add a categories display to the blog, also showing how you can easily modify existing elements within Dreamweaver.

Approximate download size: 291k

Keywords
blog,dreamweaver,dw,coldfusion,asp.net,asp,jsp,coldfusion,cf,PHP,server behaviors