Creating a Simple Blog - Part 5

By:

One of the frequent questions in the forums is "How do I create a blog"? There are many commercial blog systems out there, and many full-featured blog providers, like Blogger, but you can also create a simple blog using the standard tools of Dreamweaver. This tutorial will be equally applicable to ColdFusion, ASP, PHP, ASP.NET, or JSP because we will be using standard Dreamweaver server behaviors to create the blog. Part 5 addresses the need for a sidebar, and adds a couple of typical blog features to the sidebar, such as a category list, rss feed info, and a blogroll.

Design

So far the series has only touched on functionality. Now we'll introduce a rudimentary design. When I say "rudimentary", I really mean it. The design will be spartan and only exists to show how a design might be implemented with a sidebar in a typical blog. The design will be a simple two-column CSS design with a header and a footer. The HTML code for the page design is as follows:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<title>Blog Index</title>
<link href="css/blog.css" rel="stylesheet" type="text/css">
</head>

<body>
<div id="title"><h1>My Blog</h1></div>
<div id="sidebar">items</div>
<div id="content">yada</div>
<div id="footer">My stuff copyright 2005 by ME</div>
</body>
</html>
```

As you can see, there are 4 simple divisions of content, with ID attributes so the the CSS can be applied to the page. The blog.css file is as follows (background colors applied only to show divisions):

```
body, p, td, li {font-family:Verdana, Arial, Helvetica,
sans-serif;font-size:95%;margin:0}
#title {background-color:#FFF0F0; height:90px}
#sidebar {position:absolute; left:0; top:90px; width:180px; padding:10px;
background-color:#FFF2FF}
#content {left:0;top:90px; margin-left:200px; padding:10px;
background-color:#EEF7F7}
#footer {text-align:center; background-color:#FFFFE6}
```

Save the css file in a folder called **css** as **blog.css**, and save the main design as **simplelayout.php** (or .cfm, .asp, or whatever server language you are working with.) This page can be used as a basis for other pages, or used as a basis for a template or server-side include.

With the design in place, we'll implement it on our main blog index page.

Adding the design to the Index page

Open the index page and attach the stylesheet to the page by either typing it into code view or using Dreamweaver's tools to attach it (Design view, **Right-click > CSS Styles > Manage Styles > Attach**). Next, copy/paste the 4 div tags into the body of the document. Remove all existing content within the body tags and place it within the content div. If you save the file and view it in a browser, you should now see the blog inside the design. The page should now look like Figure 1:

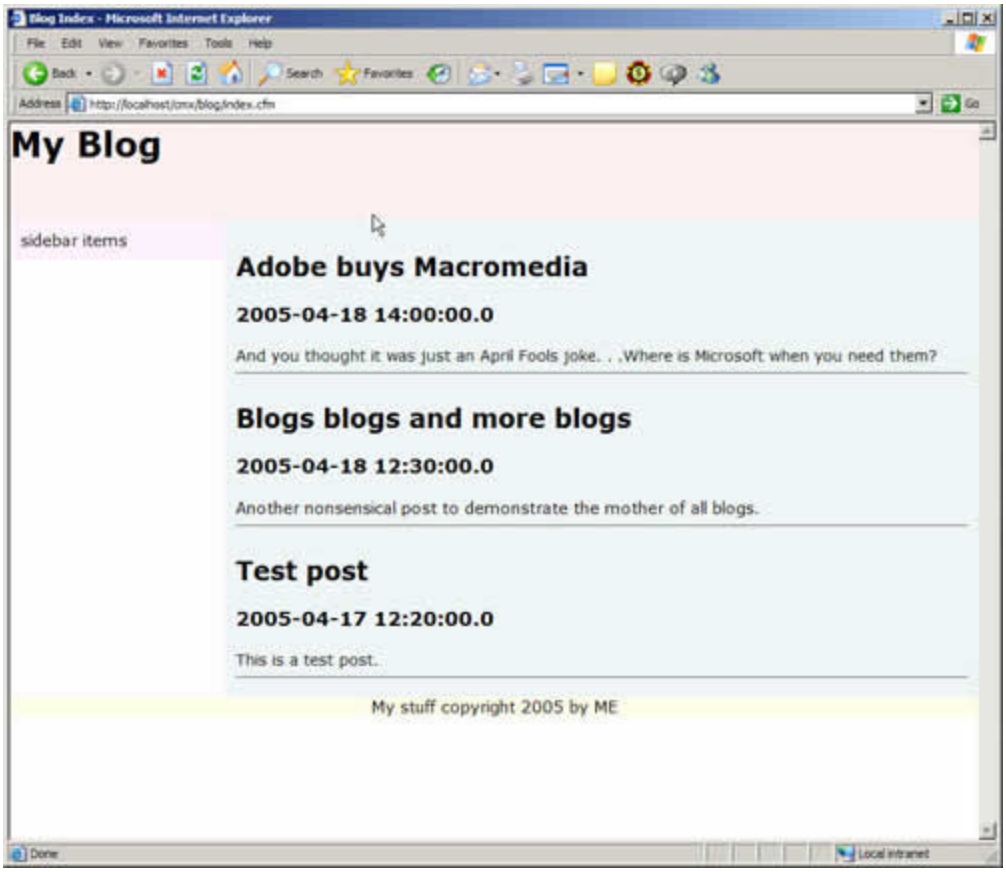

Figure 1: *Viewing the blog incorporating the new design*

Creating Modules

Modules are going to be created for each blog feature that we want to add. When I say "module", I am referring to a piece of code that is isolated in its own file, not any technical idea of a module, such as a <cfmodule> in ColdFusion. The file will be included via server-side include. This makes the blog system easy to modify and manipulate later. Each module should be completely self-contained and function by itself. In other words, if you browse to a module, you should be able to view the results.

Each module will be contained in its own div and given an ID so that you can later control the content with CSS if you so desire. It will also contain structural tags of `<h3>` to contain a heading for the module, and `<p>` tags for the content of the module. A typical module file will look like this:

```
<div id="rss">
<h3>Syndicate this blog</h3>
<p><a href="rss.php">RSS Feed</a></p>
</div>
```

And there you have your first module--a link for your RSS feed from Part 3. Save it in a subfolder called **modules** in your main blog site as **rssmodule.php**. Moving back into the **index.php** file, place an include file into the sidebar div:

PHP

```
<?php include("modules/rssmodule.php");?>
```

If you are using another server language, one of these statements will work:

ColdFusion

```
<cfinclude template="modules/rssmodule.cfm">
```

ASP/ASP.NET

```
<!--#include file="modules/rssmodule.asp" -->
```

JSP

```
<jsp:include page="modules/rssmodule.jsp" />
```

Categories Module

The next module we'll create will be a simple list of categories. The categories list will be exactly like the categories list we made for the admin page, only with a link back to a copy of the index page with a category URL variable to filter the recordset. To create the **categorymodule.php** file, first start with a skeleton:

```
<div id="categories">
<h3>Blog Categories </h3>
<p>
<a href="indexcat.php?category=[category_id field goes here]">[category field
goes here]</a><br>
</p>
</div>
```

Then add a Dreamweaver recordset to the page, using this SQL:

```
SELECT category_id, category_desc FROM blog_categories
ORDER BY category_id
```

Drag the **category** field from the Bindings panel into Code View, replacing the text [category field goes here]. Drag the **category_id** field from the Bindings panel to the page replacing the text [category_id field goes here].

Finally, create a repeat region using the server behaviors panel. Select the entire link with the `<br>` tag and apply the repeat region to the area using the recordset you just created. Save the file.

Add a server-side include to the main index page exactly as you did for the rss module. If you browse the index page now, you'll see the two modules flowing into the design.

Filtering the Blog for Categories

The category links now point back to a copy of the main index page with a URL variable passing the category id. We could also link back to the main index page and do some conditional logic on the recordset to filter by category ID, but instead we'll simply make a copy to keep the logic simple. First, copy the index.php page and save it as **indexcat.php**. Next, double-click the **rsBlogItems** recordset in the Bindings panel to modify it. You'll be able to modify it easily with these 3 simple steps:

1. Make sure you are in Simple mode. If not, click the Simple button to put the dialog into Simple mode.
2. Filter the blog_category field by choosing it in the dropdown box.
3. Filter by a URL parameter named category by typing it into the box.

Click OK to create the newly modified recordset, and save the page. If you browse it now, you should be able to click the category links in the sidebar and filter your items by category.

Note: Although we used a separate page to filter categories for this tutorial, you are encouraged to try building a one-size-fits-all index page with conditional logic for a category filter.

Blogroll

The last module we'll add for now will be a blogroll. A blogroll is a simple list of links to other blogs. Most blogs have them. It gives your blog readers a chance to view the blogs that you read regularly. Before creating the blogroll module, we'll have to add a blogroll table to the database (unrelated to the other blog tables) and add an admin page for inserting new links into the blogroll:

Field Name	Data Type	Special characteristics
blog_linkid	int	identity or autonumber
blog_linktext	varchar (128)	
blog_linkhref	varchar (255)	
blog_linkshortdesc	varchar (255)	

The SQL to create the table for the blogroll looks like this:

```
CREATE TABLE blog_blogroll (
 blog_linkid int IDENTITY (1, 1) NOT NULL ,
 blog_linktext varchar (128) NULL ,
 blog_linkhref varchar (255) NULL ,
 blog_linkshortdesc varchar (255) NULL
)
```

For MySQL:

```
CREATE TABLE blog_blogroll (
 blog_linkid int AUTO_INCREMENT NOT NULL, PRIMARY KEY(blog_linkid),
 blog_linktext varchar (128) NULL ,
 blog_linkhref varchar (255) NULL ,
 blog_linkshortdesc varchar (255) NULL
);
```

After running the scripts (or manually creating the table for Access), create an admin page as we did in Part 4 for categories.

Administering the Blogroll

We'll create a page now to add links to the blogroll. Create a page named **blogroll.php** (or .cfm, .asp, or whatever server model you are working with) in the admin section. On that page, create a recordset named **rsBlogroll**. The recordset should use this SQL to list out all the existing categories:

```
SELECT blog_linkid, blog_linktext, blog_linkhreff
FROM blog_blogroll ORDER BY blog_linktext
```

Next, add a Dynamic Table (**Insert > Application Objects > Dynamic Data > Dynamic Table**) to the page using the **rsBlogroll** 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_blogroll
- **After inserting, go to**: blogroll.php
- **Form Fields**: Use the minus button to remove the **blog_linkid** field, and set the following fields:

Column	Label	Display As	Submit As
blog_linktext	Link Text :	textfield	Text
blog_linkhref	Link:	textfield	Text
blog_linkshortdesc	Short Description:	textarea	Text

The resulting page will display all links, and also allow you to add new links to the blogroll, as shown in Figure 2:

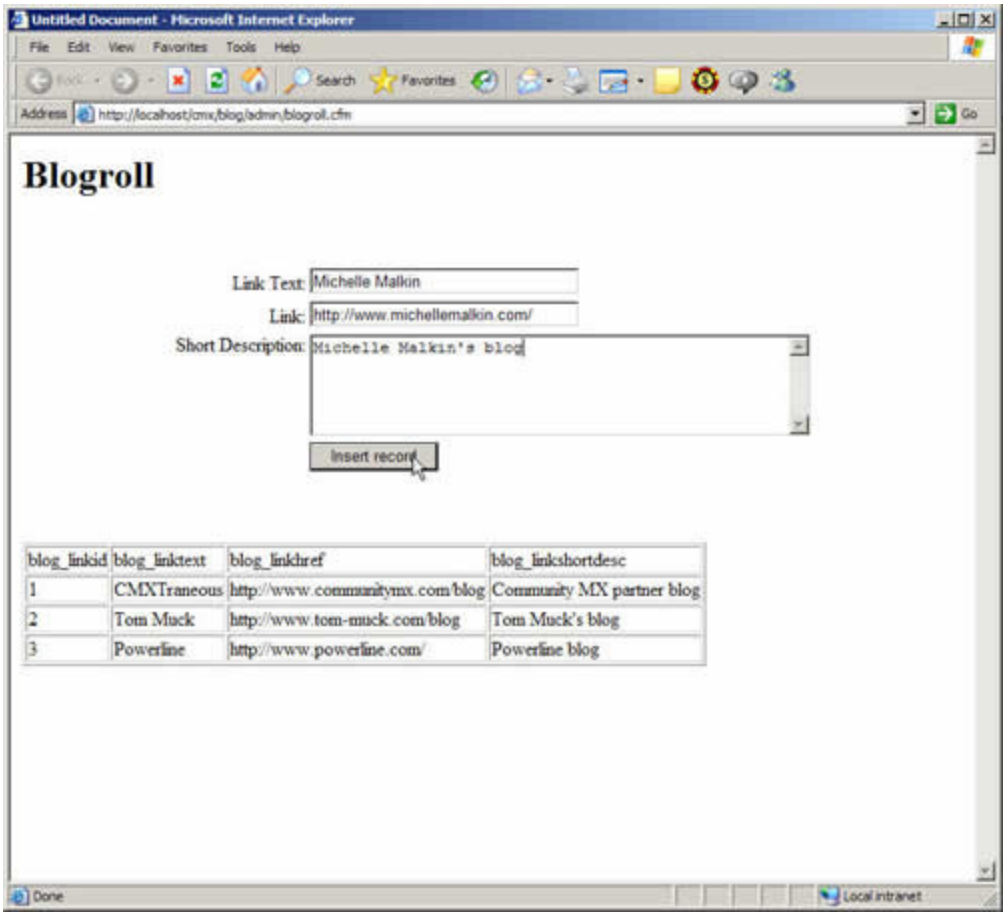

Figure 2: *The blogroll links administration page*

Try it out by browsing the page and adding a few links to your database.

Note: You can set up your own column headings for the table that lists out the records in the Blogroll table so that they are more user friendly. Dreamweaver will use the default database field headings when you use the Dynamic Table object.

You can set up an update page for the blogroll admin page as well, using the **Record Update Form Wizard** from the Application tab.

Now, we'll create the list of links for the blogroll module the same way we created the list of categories. First start with a skeleton:

```
<div id="blogroll">
<h3>Blogroll</h3>
<p>
<a href="[blog_linkhref field goes here]">[blog_linktext field goes
here]</a><br>
</p>
</div>
```

Then add a Dreamweaver recordset to the page, using this SQL:

```
SELECT * FROM blog_blogroll
ORDER BY blog_linktext
```

Drag the **blog_linkhref** field from the Bindings panel into Code View, replacing the text [blog_linkhref field goes here]. Drag the **blog_linktext** field from the Bindings panel to the page replacing the text [blog_linktext field goes here].

Finally, create a repeat region using the server behaviors panel. Select the entire link with the `<br>` tag and apply the repeat region to the area using the recordset you just created. Save the file.

Add a server-side include to the main index page exactly as you did for the rss module and the categories module. If you browse the index page now, you'll see the three modules flowing into the design, as shown in Figure 3:

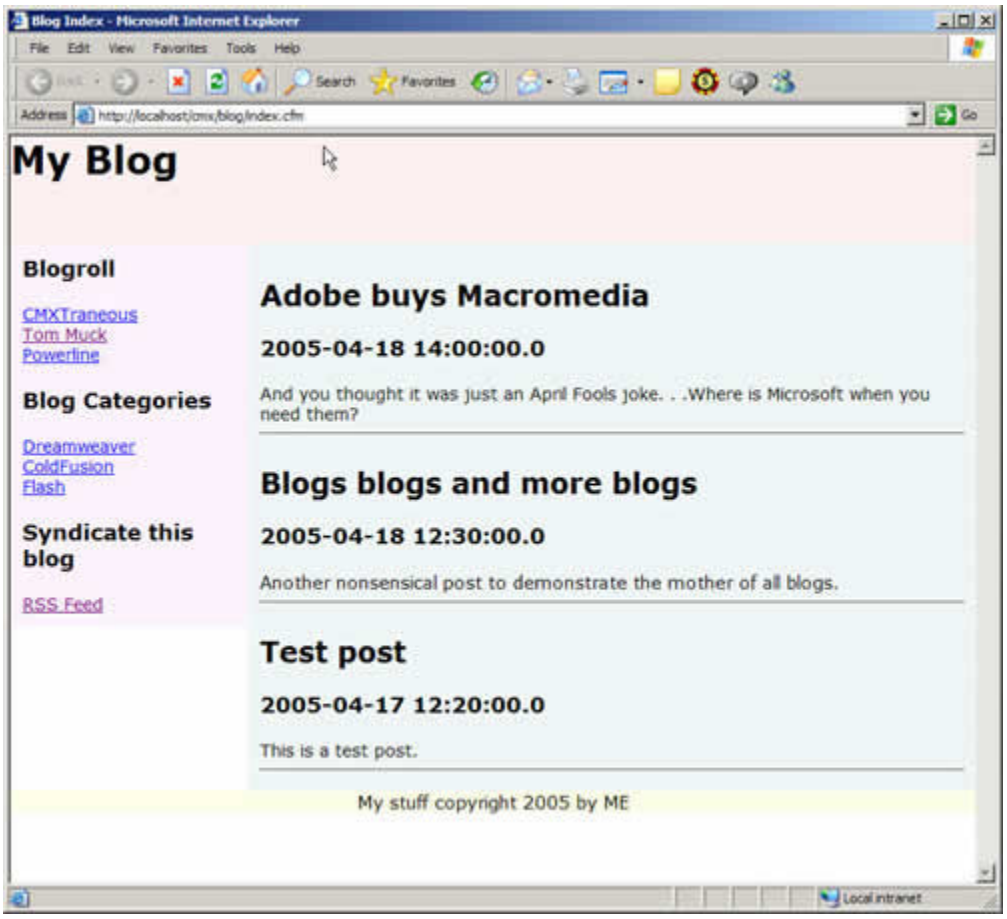

Figure 3: *Completed sidebar with 3 new modules*

Conclusion

Blogs are simple content management systems that can be put together easily using some built-in Dreamweaver functionality. This part of the series showed how to incorporate a design into the blog and use the sidebar for typical blog features like a category list and a blogroll. You can use the techniques to also create a calendar, archives list, polls, advertisements, or any other type of module that you might want to include in your blog.

Keywords
blog,dreamweaver,cf,coldfusion,asp,asp.net,jsp,php,rss