

[Home](#) / [Developer Center](#) / [Dreamweaver Developer Center](#) /

# Dreamweaver Article

## Designing with CSS in Macromedia Dreamweaver MX 2004

by **Drew McLellan**  
[Dreamweaver Fever](#)

We used to ride bicycles a great deal when I was a child. As the smallest of the three children, I inevitably ended up riding my sisters' hand-me-down bicycles, which, after a lick of paint and the removal of the odd basket, mud-guards and bells, looked quite respectable. As my sisters grew up and lost interest in bikes, the steady stream of hand-me-down bicycles depleted to a trickle, and eventually stopped all together. Before long, I could only cycle in a straight line—any attempt in changing direction bumped my knees against the handlebars, usually causing me to fall off sideways. I could adjust the bike's saddle and handlebars no more, I had extended them to their full extent—I was just too big for my bike.

For a long while up until recently, I encountered similar problems when using CSS layouts in Macromedia Dreamweaver. While Dreamweaver MX handled CSS layouts a great deal better than any previous version, the full support still wasn't there and I constantly found my knees hitting the handlebars. Keep going in a straight line and you'll be fine, but try anything a little trickier and you'd be on your own.

### CSS Improvements

Dreamweaver MX 2004 boasts a slew of improvements to its CSS capabilities that reach right across the scope of the application. From the remodeling of the primary Property inspector to a new set of panels for viewing and editing rules, CSS technology is at the program's core.

The biggest area of improvement, however, is in the rendering of pages that use CSS. Rendering is geeky word that refers to taking a set of instructions (like a web page with CSS styling) and forming a visual representation on screen. So that is to say, Dreamweaver MX 2004 displays pages using CSS a lot more accurately. This is a tremendous improvement in itself, because unless Dreamweaver can display your layout correctly, none of the other visual tools are useful. The improved rendering makes Dreamweaver MX 2004 a viable tool for working with CSS layouts, and the improved editing facilities make it viable tool for creating them too.

For a full description of the new CSS features in Dreamweaver MX, refer to Julie Hallstrom's article, [An Overview of CSS in Dreamweaver MX 2004](#).

In this article, I'll step through how to put together a CSS layout using the new tools offered in Dreamweaver MX 2004. After following this tutorial, you should have a good idea of how to start using Dreamweaver to build your own CSS based layouts.

### Requirements

#### Dreamweaver MX 2004

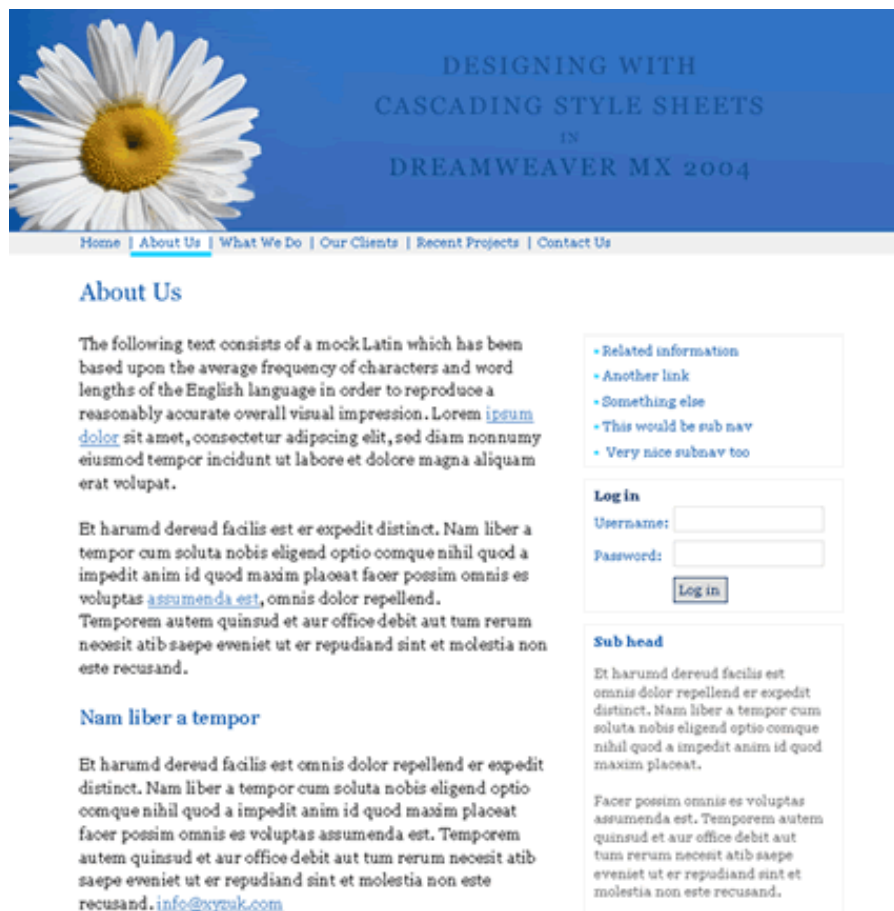
-  [Try](#)
-  [Buy](#)

## Sample files for this tutorial

-  [css\\_design.zip \(17K\)](#)

## The Design

The design we will build in this article is intended to be a fairly common design with key elements. It has a top banner graphic, main navigation area, main content area, and a side bar that contains sub-navigation and additional features (see Figure 1). These are the kind of page elements that we deal with every day. You should be able to port techniques you learn in this tutorial to your own projects easily.

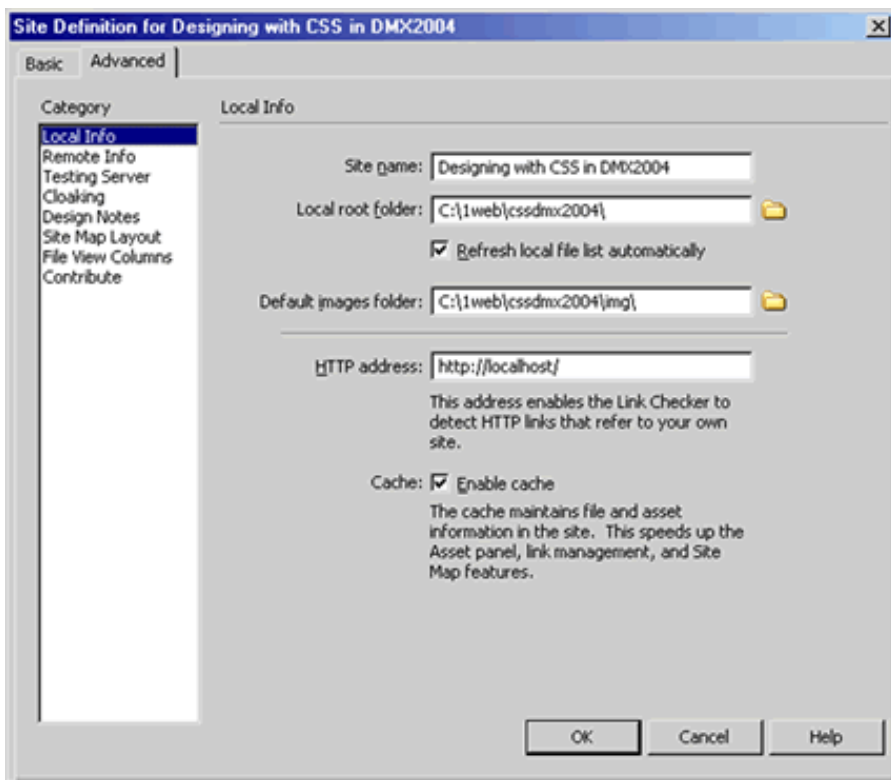


**Figure 1.** A Fireworks MX 2004 mock-up of the page we aim to build.

You're not going to be using HTML tables to lay out this page—you'll design this page with positioning properties in CSS. Historically in Dreamweaver, objects known as layers handled CSS positioning. I imagine that most people have used layers for one thing or another without realizing they were actually just using CSS. A layer is simply an HTML DIV tag with some CSS positioning applied. When working with CSS in Dreamweaver, positioned elements are displayed in just the same way as layers. So even if you haven't attempted anything of this nature before, the results should be familiar.

## Setting Up the Page

First, you must prepare the ground. In Dreamweaver MX 2004, it is no longer necessary to always define a Site Definition when working on a single file. However, since you will work with a number of files (the page, a stylesheet, and some images) it's best that you define a site for this project.



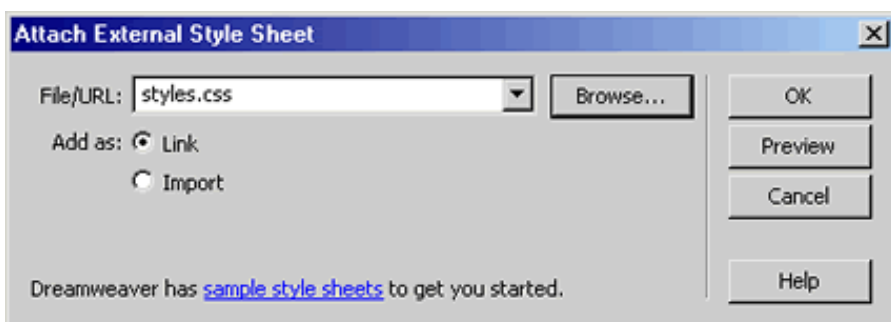
**Figure 2.** Defining a site in Dreamweaver.

Once you have defined the site, it is best to ensure that all the files you will need are in place. For this project, you'll need the following files from the sample files download at the beginning of this article. Unzip the `css_design.zip` file into the root of your site. This ZIP archive contains:

- `index_final.html`: The completed web page. I have set mine to be XHTML compliant. You will be building this page from scratch. Feel free to open it and inspect the final product if you wish.
- `styles_final.css`: The completed style sheet that will contain the layout and styling details. You will build your own style sheet in this tutorial, but you can use this as a reference.
- `img` folder: You will be using the `bgtile.jpg`, `flower.jpg`, and `titletext.gif` files in this tutorial, so make sure the `img` folder containing these files is located in the root of your site that you defined above.

Next, you'll create and link the style sheet and page using the following steps:

1. Create a new page in the root of your site and name it `index.html`.
2. Create a new style sheet in the root of your site (File > New > Basic Page > CSS > Create) and name it `styles.css`.
3. With the `index.html` page open, locate the CSS Styles panel under the Design panel group. This panel will list no choices presently, as your new page has no styles defined yet.
4. Click the Attach Style Sheet button at the bottom of the panel.
5. Browse to `style.css`.



**Figure 3.** Attach an External Style Sheet.

Now you're ready to build your page.

## Creating the Page Structure

Everyone has their own way of working; producing CSS layouts is no exception. One of the great advantages of using CSS for layout is that it enables you to separate your content from the styles that describe how that content should look. This usually results in clear, well-ordered pages. I tend to work a block at a time, first entering the mark-up, and then adjusting the positioning. When all the elements are positioned, I refine the style for the whole document with everything in place.

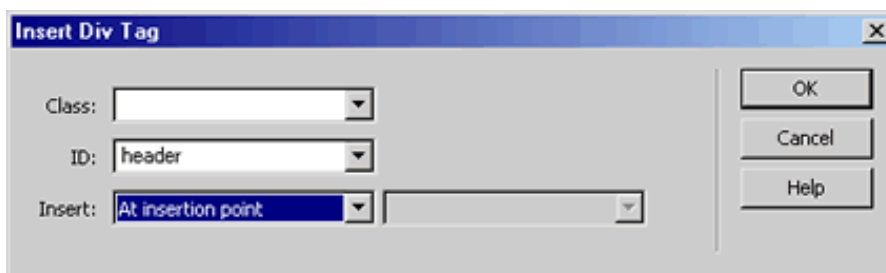
That said, I know that many people find it easier to create the page with all the blocks of content and then address the layout and styling all at once. Since this seems to be a more flowing approach, it is the workflow I will use for this article.

I find it easier to work in Dreamweaver Code view when setting up a page, since you can't see the DIV tags you'll use easily in Design view until you have positioned them. Ensure that you have opened index.html Dreamweaver and have switched to Code view before proceeding.

### The Banner

The design requires a graphical banner along the top of the page. Since the navigation bar tucks neatly under the banner, I will wrap two elements in the same container, grouping them so I can deal with them as one element.

1. Place the cursor between the BODY tags. Open the Layout menu of the Insert bar and click the new Insert Div Tag object.



**Figure 4.** Insert DIV Tag

2. For your DIV tag, select an ID of **header** and click the OK button to add the DIV tag to the page. This tag is the container for the graphical banner and the main navigation bar. The code inserted into the document looks like the following:

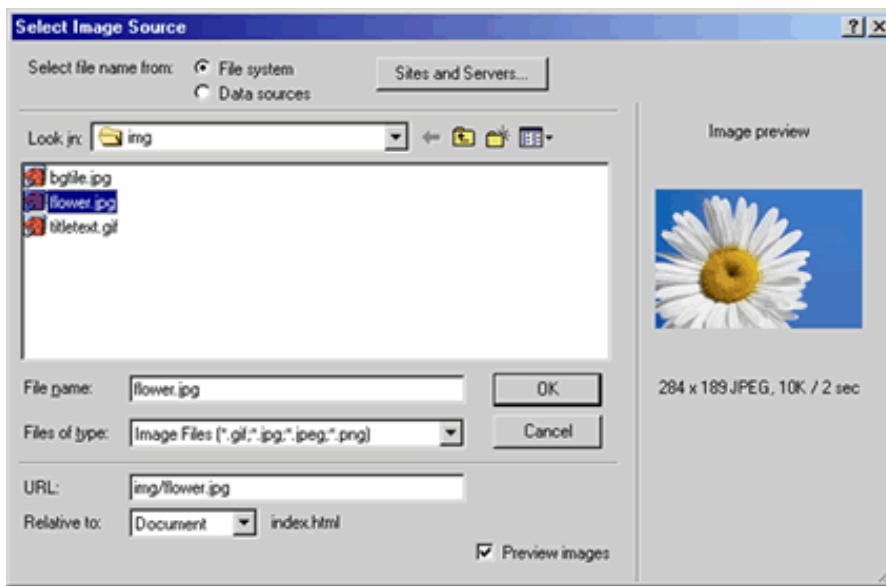
```
<div id="header">Content for id "header" Goes Here</div>
```

Dreamweaver has inserted some placeholder text inside the DIV tag that you can now delete. The different id values in the id drop-down list are being pulled from the styles.css file that you linked to in the previous section.

3. Put your cursor within the tag and click the Insert Div Tag and insert a new tag with an ID of **banner**. Click OK to add the DIV tag to the page. Again, delete the placeholder text.
4. Place the cursor just before the last closing DIV tag and click Insert Div Tag to create a tag with an ID of **nav**. Delete the placeholder text. The resulting code should look like the following:

```
<div id="header"><div id="banner"></div> <div id="nav"></div> </div>
```

The structure of the tags shows that you have one header section which contains the graphical banner and the main navigation. My design uses two images in the banner: the first is a flower and the second is a text graphic. Place your cursor within the banner div tag, and insert your images into the banner using Insert Image from the Common menu on the Insert bar.



**Figure 5.** Insert the images.

5. You may need to switch into Design view to configure ALT attributes and the image dimensions. I've also given my DIV tags ID attributes so that I can easily reference them in my CSS.

```
 
```

6. Put your cursor within the nav DIV tag and place some links inside the navigation bar. I placed my links inside an unordered list. By default, the HTML UL tag presents each item on its own line, each with its own bullet mark. However, with CSS you can redefine how you want the list to look; the benefit is that this technique will present a neat list when a primitive device like a cell phone views the page. The code for my links is as follows:

```
<div id="nav"> <ul> <li><a href="/" title="Home">Home</a></li> <li> | <a href="/about/" title="About Us">About Us</a></li> <li> | <a href="/what/" title="What We Do">What We Do</a></li> <li> | <a href="/clients/" title="Our Clients">Our Clients</a></li> <li> | <a href="/projects/" title="Recent Projects">Recent Projects</a></li> <li> | <a href="/contact/" title="Contact Us">Contact Us</a></li> </ul> </div>
```

7. Just as you have done before, create two more DIV tags below the last closing DIV tag. One contains the main content and one contains the side bar. Use Insert Div Tag button within the Insert bar's Layout category as you did before and remove the placeholder text that Dreamweaver inserts within the DIV tags.

```
<div id="content"></div> <div id="sidebar"></div>
```

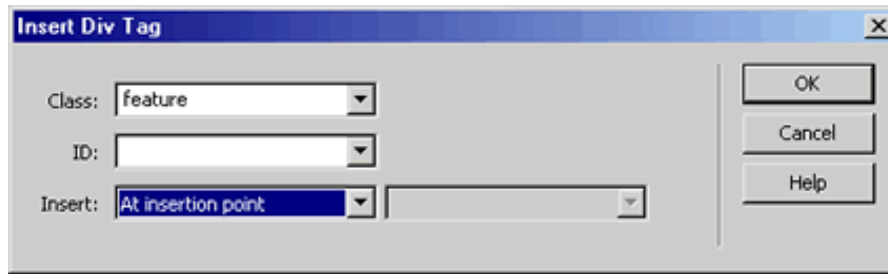
8. Add the following content within the content DIV tags:

```
<h1>About Us</h1> <p>The following text consists of a mock Latin which has been based upon the average frequency of characters and word lengths of the English language in order to reproduce a reasonably accurate overall visual impression. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy eiusmod tempor incididunt ut labore et dolore magna aliquam erat volupat.</p> <p>Et harumd dereud facilis est er expedit distinct. Nam liber a tempor cum soluta nobis eligend optio comque nihil quod a impedit anim id quod maxim placeat facer possim omnis es voluptas assumenda est, omnis dolor repellend. Temporem autem quinsud et aur office debit aut tum rerum necesis atib saepe eveniet ut er repudiand sint et molestia non este recusand.</p> <h2>Sub head</h2> <p>Et harumd dereud facilis est omnis dolor repellend er expedit distinct. Nam liber a tempor cum soluta nobis eligend optio comque nihil quod a impedit anim id quod maxim placeat facer possim omnis es voluptas assumenda est. Temporem autem quinsud et aur office debit aut tum rerum necesis atib saepe eveniet ut er repudiand sint et molestia non este recusand.</p>
```

9. Within the sidebar, I want to display featured content in boxes. Therefore, use the Insert Div Tag again to create the feature boxes. However, this time, you must assign the DIV tags a Class rather than an ID attribute. Insert three DIV tags for good measure.

**Note:** An object's ID is always unique within the document. No two items can share the same ID. A Class, however, can

appear one or more times and you can apply it to the same rules to multiple items. You may have noticed that the ID identifiers were removed from the ID drop-down list in the Insert DIV Tag dialog box after you used them.



**Figure 6.** Inserting a DIV with a Class applied

This final step completes the main structure of the page. You are ready to pad the page with content and finally, move on to positioning and styling the individual elements.

## Inserting Placeholder Content

When you lay out a page, it helps a lot if you pad the main elements with content. The content doesn't matter, it just gives you an idea of how the content will fill the page. Many people use mock Latin; you can see samples at [Lipsum.com](http://Lipsum.com).

It is also beneficial to have samples of the different heading levels that might display on your page. The ideal way to implement headings is to use the H1-H6 tags, which not only give your content more meaningful structure, but are easy to style with CSS.

Similarly, with the sidebar, it helps to use placeholder content. I'm going to use a sub-navigation list in the first of the three feature DIV tags, as follows:

```
<div class="feature"> <ul> <li><a href="#">Home</a></li> <li><a href="#">Company History</a></li> <li><a href="#">Mission Statement</a></li> <li><a href="#">Our Offices</a></li> <li><a href="#">Awards</a></li> <li><a href="#">Job Vacancies</a></li> </ul> </div>
```

I'm also going to use a log-in form containing a table with three rows and two columns in the second of the three feature DIV tags:

```
<div class="feature"> <form method="post" action="#" id="loginform"> <table> <tr> <th><label for="username">Username:</label></th> <td><input type="text" name="username" tabindex="1" /></td> </tr> <tr> <th><label for="password">Password:</label></th> <td><input type="password" name="password" tabindex="2" /></td> </tr> <tr> <td></td> <td><input class="submit" type="submit" value="Log in" tabindex="3" /></td> </tr> </table> </form> </div>
```

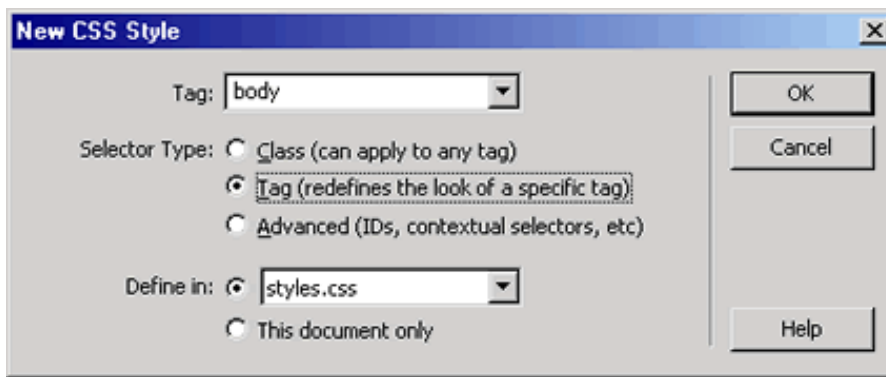
And finally, I'm going to use a short featured content paragraph in the final feature DIV tag:

```
<div class="feature"> <h1>Sub head</h1> <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
```

## Laying Out the Main Components

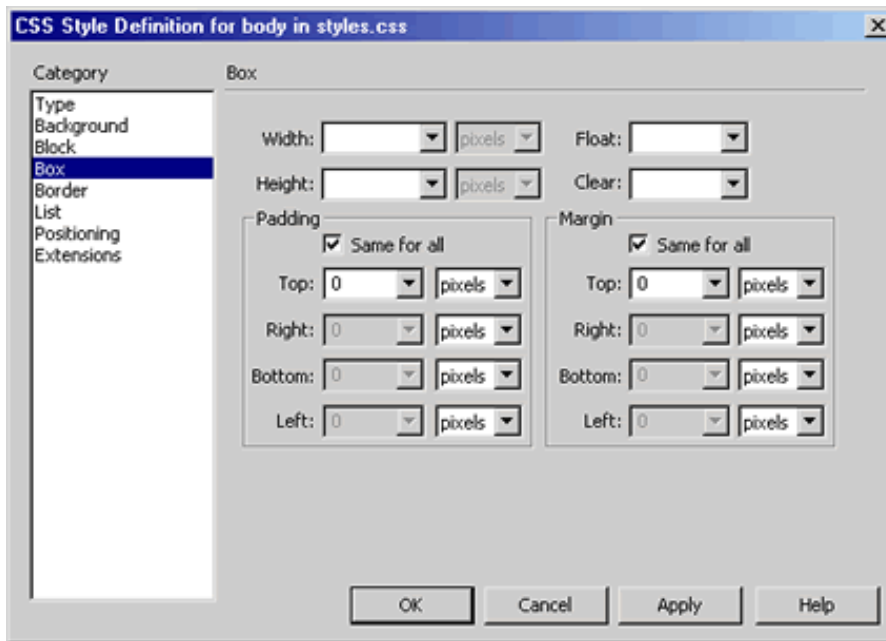
Now the fun begins! You will now start laying out elements using CSS.

1. Turn off all the default borders on the page to ensure all calculations are taken right from the edge.
2. Click the New Style button in the CSS Styles panel.



**Figure 7.** Redefining the BODY tag

3. Select the Tag selector type, and choose **body** from the list of tags (Figure 7). Ensure that you have selected your style sheet in the Define In dialog box. Click OK.



**Figure 8.** Box properties for the BODY tag

4. Select the Box category from the list and set both the Padding and Margin settings to be zero. This eliminates the default gap around the edge of the screen that you used to get rid of the four horsemen of the HTML apocalypse:

```
<body topmargin="0" leftmargin="0" marginheight="0" marginwidth="0">
```

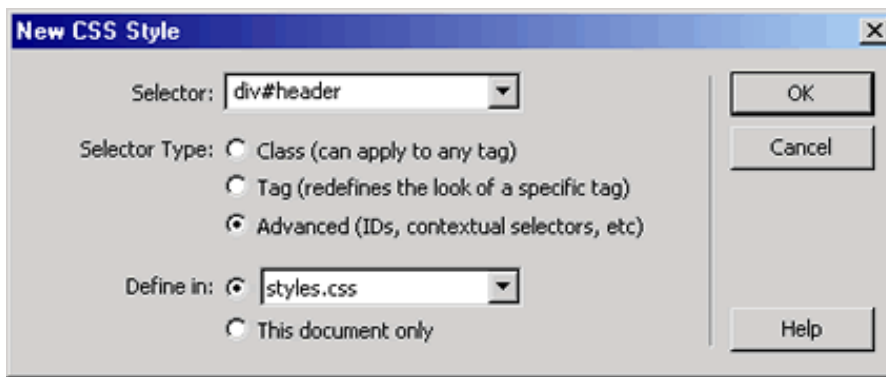
5. Click OK to accept the changes.

## The Header Block

Now you'll create the header block.

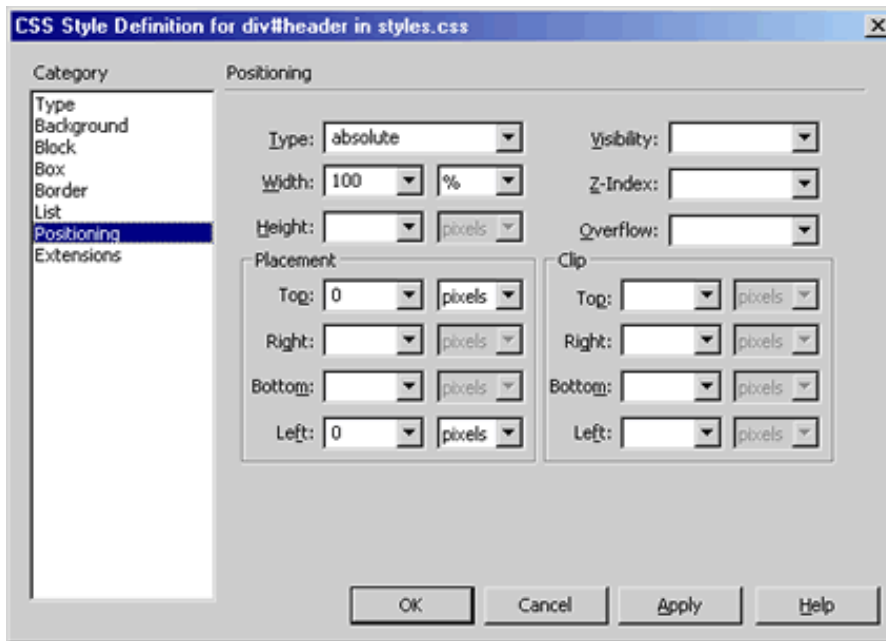
1. In the CSS Styles panel, click New CSS Style.





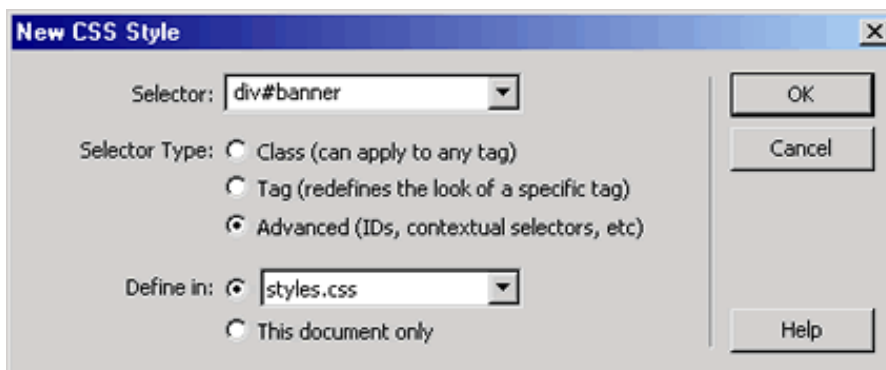
**Figure 9.** Defining the header block

2. Select the Advanced selector type, type a name of **div#header** (see Figure 9), and click OK to open the CSS Styles Definition dialog box. This name simply means "a div tag with an ID of header." The hash specifies the ID.



**Figure 10.** Defining header positioning

3. In the Positioning category, configure your settings to match those in Figure 10 and click OK to accept the changes. You use absolute positioning to take the header out of the document flow and move it to site snugly in the top left corner. As the header is now no longer in the document flow, the rest of the page will move up to occupy the space underneath it. This will leave your page looking messy for the time being, but don't worry—it's supposed to look like that, at least, for the moment!
4. Create a new style for the banner.

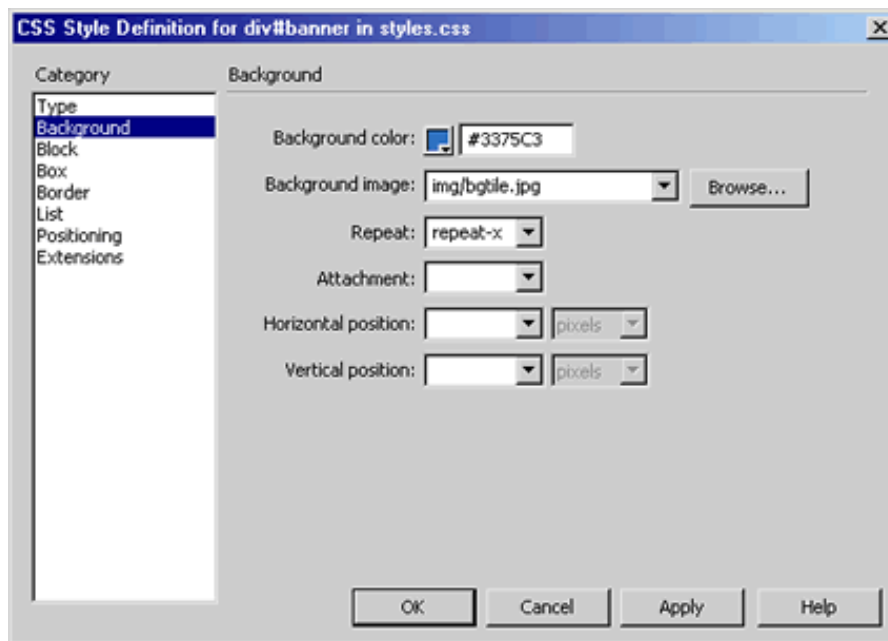


**Figure 11.** Defining the banner

This time, select the Background category. This is where you will set the background color, image, and repeat style using the

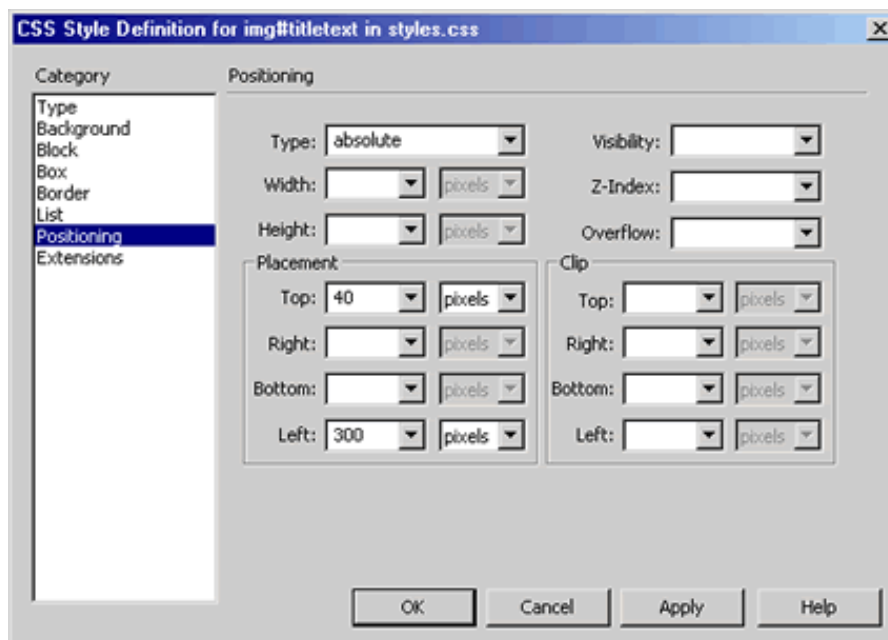


values in figure 12 below. Click OK to accept the changes.



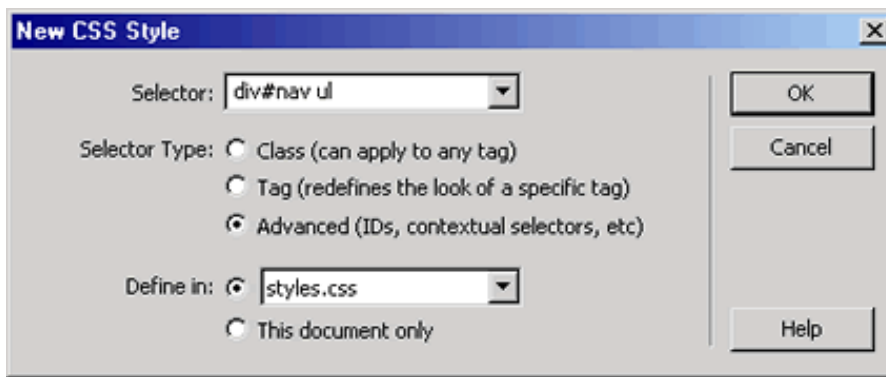
**Figure 12.** Defining the background properties of the banner

5. A Repeat setting of repeat-x tells the browser to tile the background in one direction, along the x-axis. For those who don't remember back to school mathematics, the x-axis flows from left to right.
6. To place the text graphic, create a new style called **img#titletext**, which means an IMG tag with the ID of titletext. I set it to use absolute positioning, with appropriate values to put it roughly the right area. You can change this easily, later.



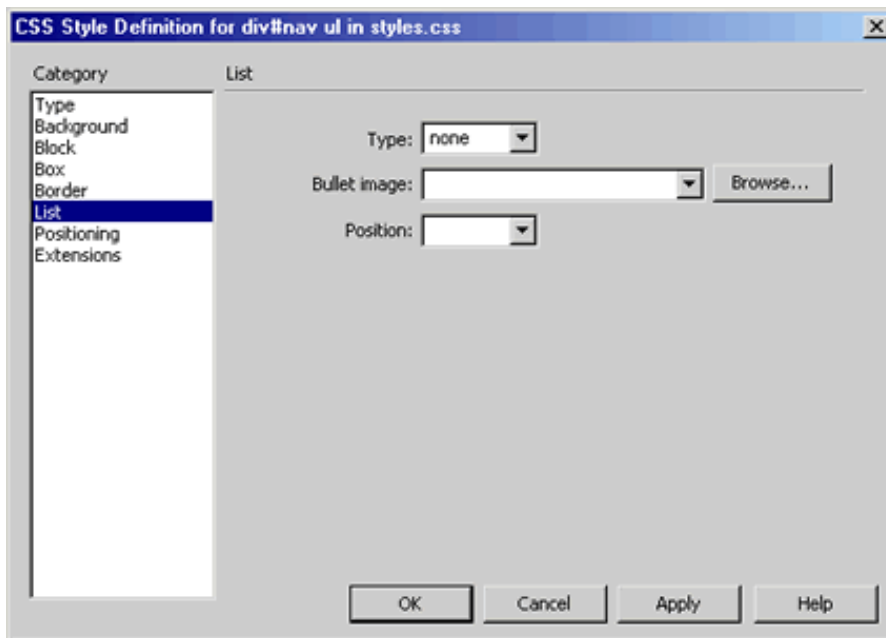
**Figure 13.** Positioning the title text image

7. To create a functioning header, you must reformat that navigation list. The selector rules required for this task are a little more complex than those we've used so far. The first rule specifies that the unordered list should use a list style of "none." The second rule specifies that all the items within the unordered list should appear on one line, without line breaks for each item.



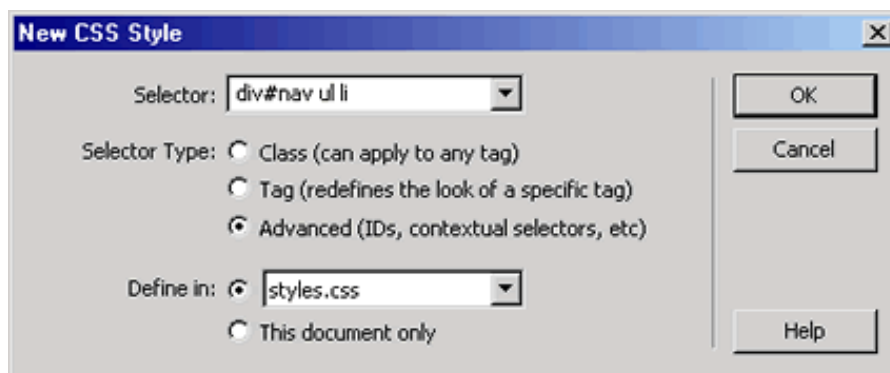
**Figure 14.** Defining the navigation list

For the selector, type **div#nav ul**, which simply means any unordered list tag within a DIV tag with the ID of "nav."



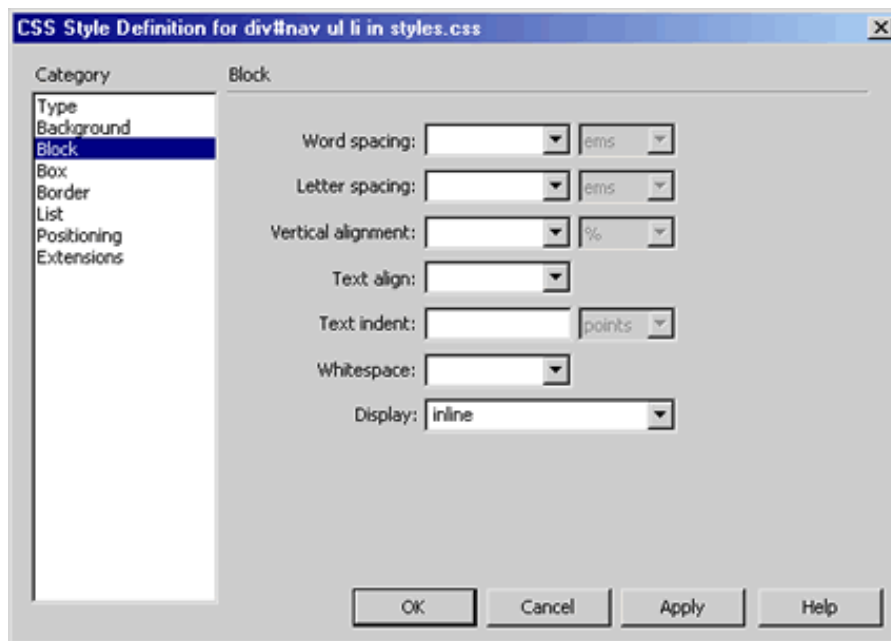
**Figure 15.** Setting list style to 'none'

1. Setting the list style to 'none' prevents any bullet marks displaying.



**Figure 16.** Defining list items within the navigation list

2. Specify settings for the list items within the navigation list. For the selector, type **div#nav ul li**, which means that any list item within an unordered list in a DIV tag with the ID of "nav." You must be getting the hang of reading these by now!



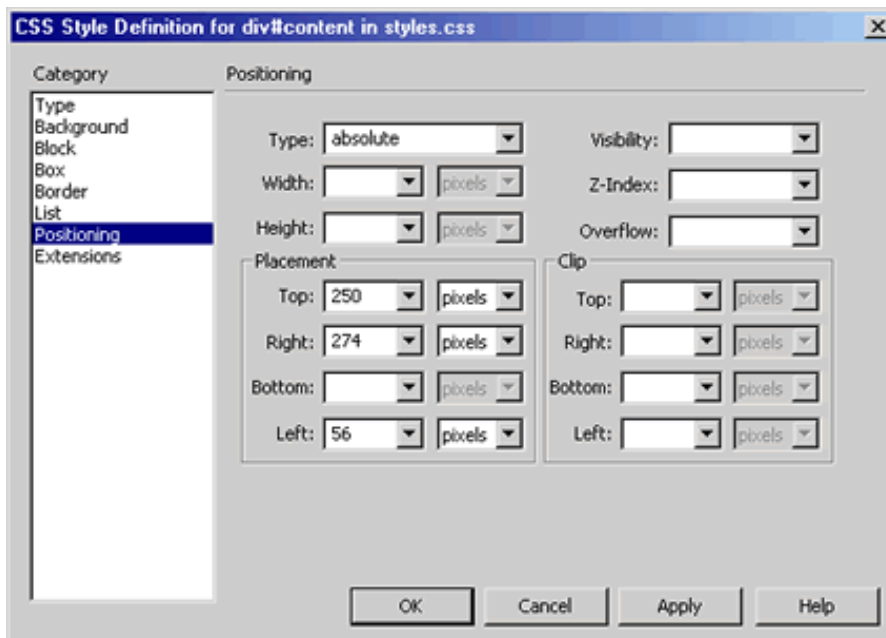
**Figure 17.** Setting list items to display inline

3. Under the Block category, for Display, select inline, which forces list items to display **inline**. This will make the list look less like a list and more like a navigation bar.

## The main content and sidebar

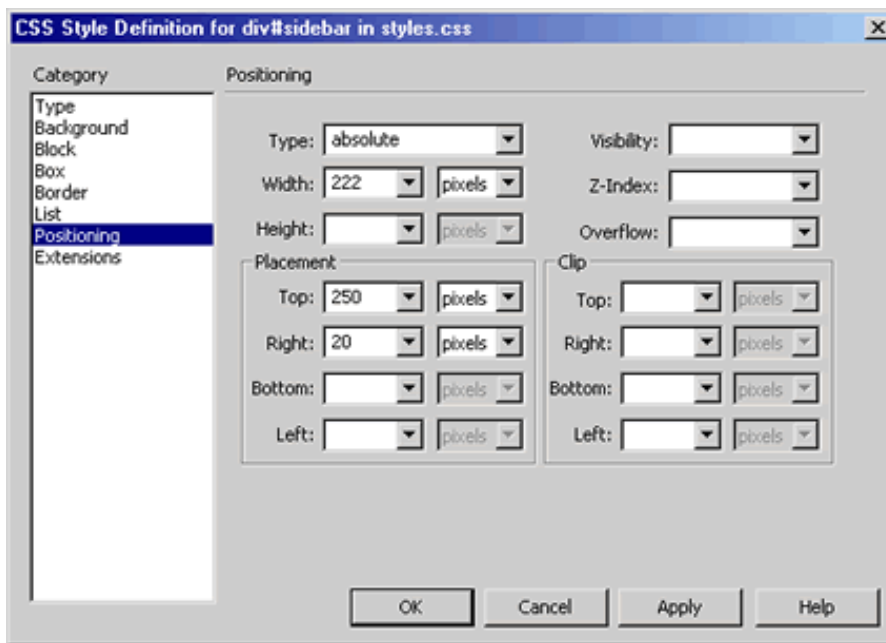
Using the techniques you are now familiar with, you can position the main content, and sidebar.

1. Settings for the main content, or **div#content** (see Figure 18):



**Figure 18.** Positioning settings for the main content

2. Settings for the sidebar, or **div#sidebar** (see Figure 19):



**Figure 19.** Positioning settings for the sidebar

It's important to note that all you're doing at this point is putting the elements into approximate positions so that you can see a mockup of your design. You will tweak and fine tune the design later to create the exact layout.

If you look at the layout in a browser, you can see that it already looks good.



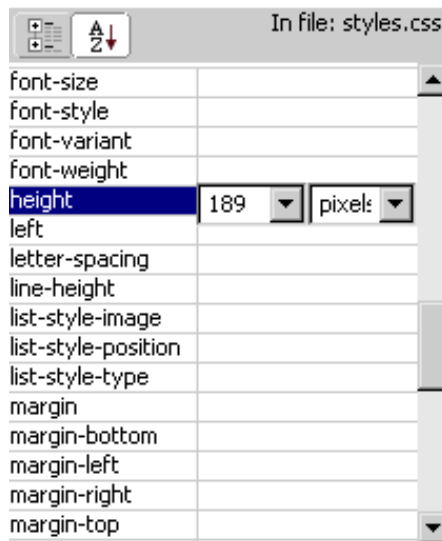
**Figure 20.** Viewing the layout in a browser (Mozilla Firebird)

## Fine-Tuning the Layout

Just as you created CSS styles in the CSS Styles panel, you will modify styles in the Relevant CSS panel. You can find the Relevant CSS panel in the Tag inspector panel group.

The Relevant CSS panel is great for making all the iterative tweaks that are required for creating a layout. Take a look at the screen shot of the page ([Figure 20 in the previous section](#)). Those with sharp eyes will spot a thin stripe of blue underneath the flower image that shouldn't be there. This is because you must fix the height of the banner.

Ensure that you have the Relevant CSS panel open (Window > Tag Inspector) and then click somewhere within the banner DIV. You'll see that the Relevant CSS panel reloads to show the styles that apply to the banner. Simply locate the `height` property (it's in alphabetical order by default) and set it to a fixed value.



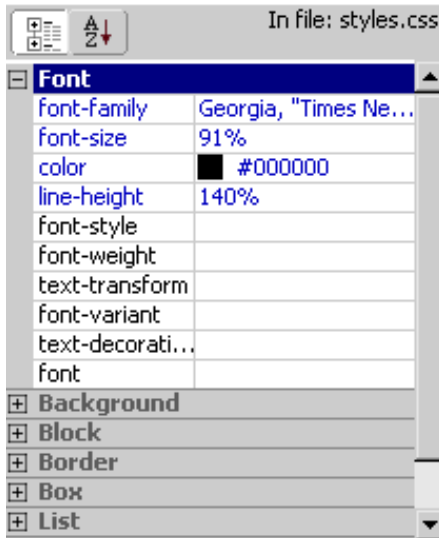
**Figure 21.** Using the Relevant CSS panel to tweak the height of an object

Once you have defined the main styles, they all appear in the Relevant CSS panel. The panel lists all the properties available that you can set and shows you already defined values (Figure 21).

## Styling the Content

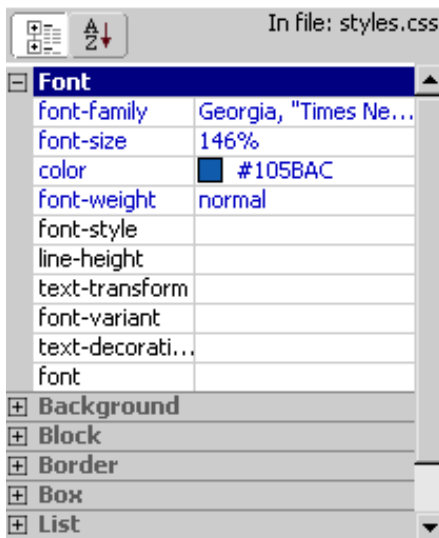
With all the elements positioned roughly where they're needed, you can start styling the content.

1. Using the Relevant CSS panel, select the content DIV and set the fonts and sizes (Figure 22).



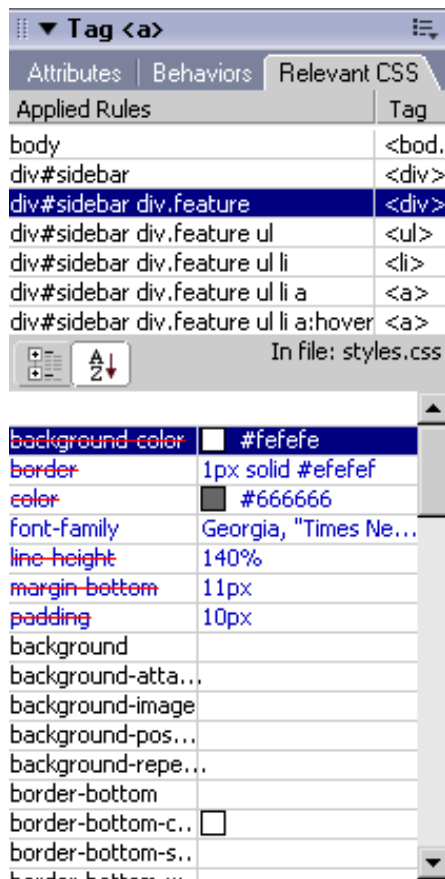
**Figure 22.** Defining font styles for the main content

- Using the CSS Styles panel, create new rules for both H1 and H2 tags within the main content and set up styles for those too (Figure 23).



**Figure 23.** Styling heading levels

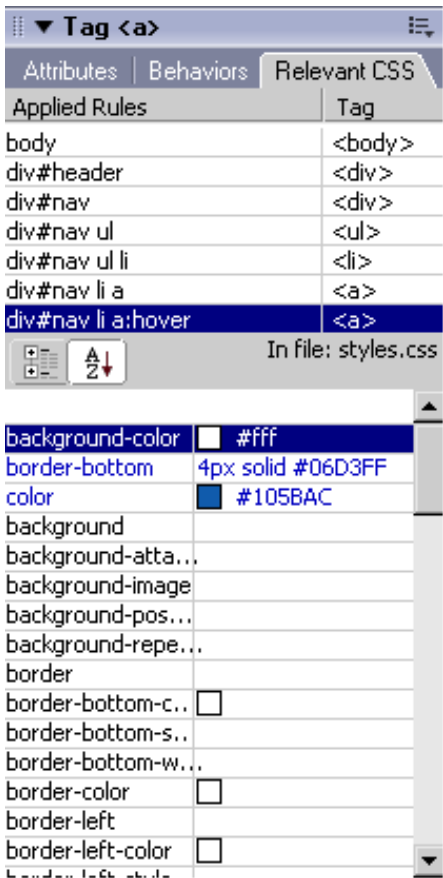
3. Similarly, define styles for the feature boxes in the side bar and edit them within the Relevant CSS panel (Figure 24).



**Figure 24.** Styling the feature boxes

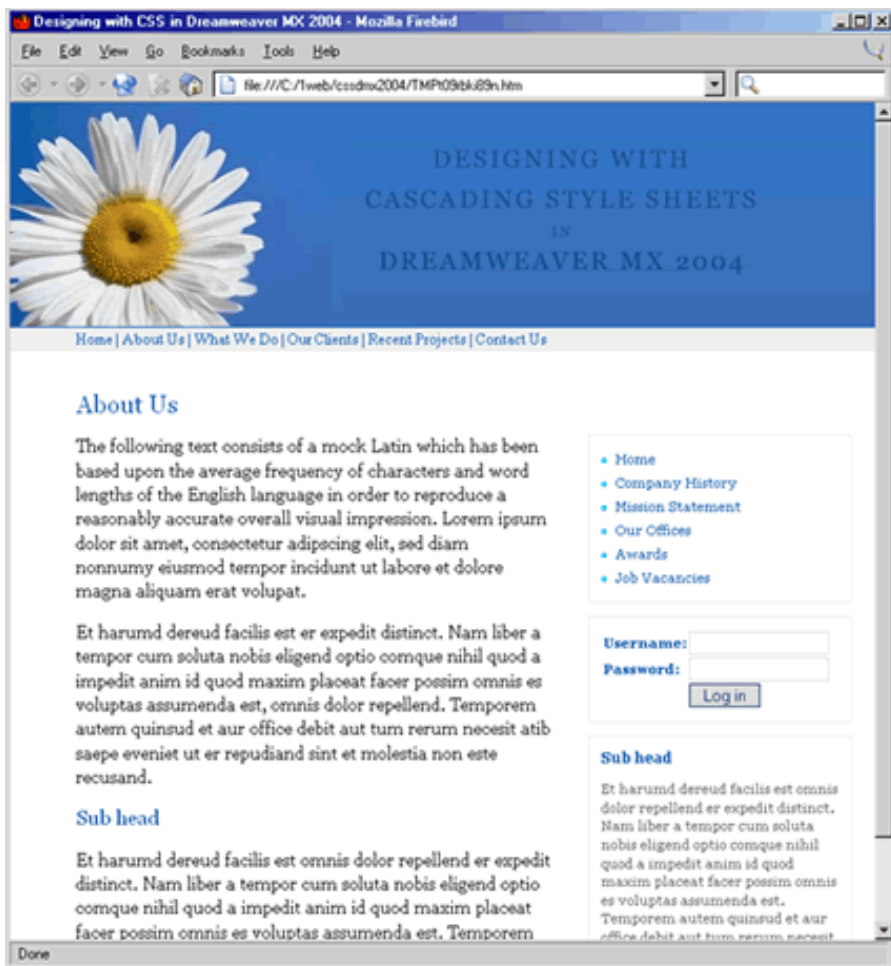
With that done, the only remaining part of the page without styles is the main navigation. You have designed an unordered list that displays in a flat line, but you haven't yet made it easy on the eye.





**Figure 25.** Defining styles for the main navigation

With that, you've completed the layout. Check your work in as many browsers and platforms as you have access to and adjust as necessary. Here's how my final page looks.



**Figure 26.** The completed page viewed in a browser

In this article, we have looked at some of the new CSS features and capabilities of Dreamweaver MX 2004. Using a knowledge of CSS and the editing tools provided, we have produced a light-weight, efficient layout that will work in multiple browsers. My hope is that after following this article (don't be afraid to read it a few times over), you will be able to take the skills you have learned and apply them to your own projects.

## About the author

[Drew McLellan](#) is the author of [Dreamweaver MX Web Development](#) from New Riders and is a member of the [Web Standards Project's](#) Dreamweaver Task Force, an extension writer, and an all-around Dreamweaver good guy. He runs [DreamweaverFever.com](#) as well as a successful web development and standards-based weblog at [allinthehead.com](#). In the real world he's a feet-on-the-ground Senior Web Developer for a dot.com business on the outskirts of London.



[Company](#) | [Site Map](#) | [Privacy & Security](#) | [Contact Us](#) | [Accessibility](#) | [Report Piracy](#) | [Send Feedback](#)

©1995-2003 Macromedia, Inc. [All rights reserved.](#)

Use of this website signifies your agreement to the [Terms of Use](#).

Search powered by 